(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2004-509539
(P2004-509539A)
(43)公表日　平成16年3月25日(2004.3.25)

(51)Int.Cl.[7]	F I			テーマコード（参考）
H04L 12/28	H04L	12/28	310	5B085
G06F 13/00	G06F	13/00	353C	5B089
G06F 15/00	G06F	15/00	310D	5K033

審査請求 未請求　予備審査請求 有　（全 248 頁）

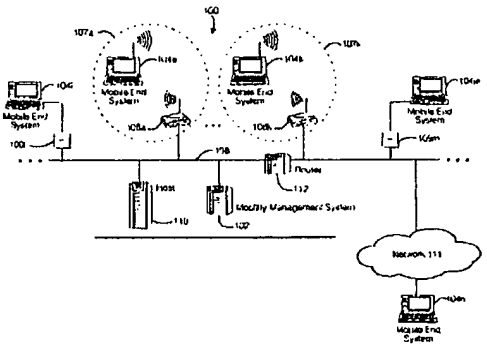
(21)出願番号	特願2002-527943(P2002-527943)
(86)(22)出願日	平成13年9月12日(2001.9.12)
(85)翻訳文提出日	平成15年3月11日(2003.3.11)
(86)国際出願番号	PCT/US2001/028391
(87)国際公開番号	WO2002/023362
(87)国際公開日	平成14年3月21日(2002.3.21)
(31)優先権主張番号	09/660,500
(32)優先日	平成12年9月12日(2000.9.12)
(33)優先権主張国	米国(US)
(31)優先権主張番号	60/274,615
(32)優先日	平成13年3月12日(2001.3.12)
(33)優先権主張国	米国(US)

(71)出願人　503095055
ネットモーション　ワイヤレス　インコーポレイテッド
アメリカ合衆国，９８１０９－３０３２　ワシントン州，シアトル，デクスター　アベニュー　ノース　１５００
(74)代理人　100080034
弁理士　原　謙三
(74)代理人　100113701
弁理士　木島　隆一
(74)代理人　100116241
弁理士　金子　一郎

最終頁に続く

(54)【発明の名称】コンピュータ環境におけるモバイル他の断続的接続性を提供する方法および装置

(57)【要約】
ノマディックなシステムの特性を透過的にする、シームレスなソリューションであって、既存のネットワーク・アプリケーションが、モバイル環境でも確実に動作するようにする。モバイルネットワークと組み合わされたモビリティ管理サーバ（１０２）は、数量を限定されないモバイル端末システム（１０４）それぞれの状態を維持し、ネットワークへ、および他のピアでの処理への持続的な接続の維持に必要な、複雑なセッション管理を実行する。モバイル端末システムが圏外に出たり、サスペンドしたり、（例えば、あるネットワーク相互接続から別のものへとローミングして）ネットワークアドレスを変更したりした場合、モビリティ管理サーバは、結合しているピアのタスクとの接続を維持し、モバイル端末システムとネットワーク媒体とのコンタクトが一時的に失われても、モバイル端末システムが持続的な接続を維持できるようにする。インターフェイスベースのリスナは、ネットワーク・インターフェイスからのネットワーク接続ポイント情報を利用して、ローミング状態を判定し、ローミングに際した接続を効率的に確立する。モビリテ



【特許請求の範囲】
【請求項１】
ネットワーク接続ポイントを介してネットワークと接続されるモバイルコンピュータ装置を少なくとも一つ含むモバイル・コンピューティング・ネットワークであって、モバイルコンピュータ装置のロケーションを含む様々なメトリクスに基づくポリシー管理ルールを適用するポリシー管理手法を特徴とするモバイル・コンピューティング・ネットワーク。
【請求項２】
前記ルールの属性についての処理は、モバイルコンピュータ装置あるいはモビリティ管理サーバのいずれか、またはその両方に対して分配され適用されることを特徴とする請求項１記載のネットワーク。
【請求項３】
前記ルールの優先順位は、そのようなテーブルの項目における位置で暗示されるか、あるいは期待動作を確実にするための順序が明白に記されることを特徴とする請求項１または２記載のネットワーク。
【請求項４】
前記ルールの属性のデータ保存は、中央管理サービスを介して局所的または集中的に管理されることを特徴とする請求項１、２または３記載のネットワーク。
【請求項５】
特定のアプリケーションの動作は、サービス費用、ネットワーク接続ポイント、信頼関係等を含む多数のメトリクスに基づいて修正されることを特徴とする請求項１～４のいずれか１項に記載のネットワーク。
【請求項６】
動作修正の結果、前記ルールの属性に基づいて要求が許可、拒否、または遅延されることを特徴とする請求項１～５のいずれか１項に記載のネットワーク。
【請求項７】
アプリケーションが既に開始されている場合でも、アプリケーションプロセスを修正するため、１つのルール、または一連のルールが呼び出されることを特徴とする請求項１～６のいずれか１項に記載のネットワーク。
【請求項８】
現在位置情報（ロケーション）は、アプリケーション動作を管理するため、あるいはモバイルコンピュータ装置の関連情報を提供するためにも使用されることを特徴とする請求項１～７のいずれか１項に記載のネットワーク。
【請求項９】
距離測定に伴う動作速度は、アプリケーションの動作または通信パスを変更するために使用されることを特徴とする請求項１～８のいずれか１項に記載のネットワーク。
【請求項１０】
ロケーション情報の結果としてトポロジ情報が抽出され、表示されることを特徴とする請求項１～９のいずれか１項に記載のネットワーク。
【請求項１１】
ネットワーク接続ポイントを介してネットワークと接続されるモバイルコンピュータ装置を少なくとも一つ含むモバイル・コンピューティング・ネットワークにおいて、ネットワーク接続ポイントの識別子の少なくとも一部分に基づき、前記モバイルコンピュータ装置が異なるネットワーク・セグメントに移動したかどうか検出するインターフェイス補助ローミングリスナを備えることを特徴とするモバイル・コンピューティング・ネットワーク。
【請求項１２】
前記モバイルコンピュータ装置はネットワーク・インターフェイス・アダプタを含み、前記インターフェイス補助ローミングリスナは前記ネットワーク・インターフェイス・アダプタから前記ネットワーク接続ポイントの識別子を取得することを特徴とする請求項１１記載のネットワーク。

【請求項１３】
前記インターフェイス補助ローミングリスナは、前記ネットワーク接続ポイントの識別子をネットワーク接続についての詳細情報に相関させる情報を記憶するネットワーク・トポロジ・マップを保持することを特徴とする請求項１１記載のネットワーク。
【請求項１４】
前記インターフェイス補助ローミングリスナは、前記ネットワークとの通信がいつ中断または再確立されるかを検出することを特徴とする請求項１１記載のネットワーク。
【請求項１５】
前記インターフェイス補助ローミングリスナは、（ａ）ネットワーク通信の中断及び再確立、及び（ｂ）前記ネットワーク接続ポイントの識別子の変化の検出を受けて、ローミング信号を生成することを特徴とする請求項１４記載のネットワーク。
【請求項１６】
モバイルコンピュータ装置において使用されるインターフェイスベースのリスナであって、インターフェイスベースのリスナは少なくとも１つのネットワーク・インターフェイス・アダプタからの情報と、少なくとも１つのネットワークスタックから入手可能な情報を統合して、前記モバイルコンピュータ装置が新しいネットワーク接続ポイントに移動したかどうか判定することを特徴とする、インターフェイスベースのリスナ。
【請求項１７】
ネットワーク接続ポイント情報を含むネットワーク接続情報を提供するネットワーク・トポロジ・マップを含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項１８】
上記リスナは学習情報に基づき前記ネットワーク・トポロジ・マップを動的に構成することを特徴とする請求項１７記載のリスナ。
【請求項１９】
イベント発生に基づき、ステータスをチェックするステータスチェッカーを含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項２０】
前記イベントはタイマーのタイムアウト、ローレベルのローミング・ドライバ・コールバック、ネットワーク・レベル・アクティビティ・ヒントのいずれかを含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項２１】
モバイルコンピュータ装置が既に現在のネットワーク接続ポイントを訪れたかどうかインターフェイスに照会する接続情報探索部を含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項２２】
新規のネットワーク・セグメントにおいて有効となる現行のアドレスを登録または再取得する接続構成を含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項２３】
インターフェイスから提供された情報の少なくとも一部分に基づいて、前記モバイルコンピュータ装置が異なるネットワーク・セグメントに移動したことの検出を受けてローミング信号を生成するローミング信号発生部を含むことを特徴とする請求項１６記載のインターフェイスベースのリスナ。
【請求項２４】
予め割り当てられたアドレスがまだ有効かどうかを判定するヒューリスティック・アナライザをさらに含むことを特徴とする請求項２３記載のインターフェイスベースのリスナ。
【請求項２５】
モバイルノードが新しいネットワーク接続ポイントに移動したかどうか判定する方法であって、

（ａ）ネットワーク・インターフェイスからネットワーク接続ポイントの識別情報を受け取るステップと、
（ｂ）前記ネットワーク接続ポイントの識別情報を使用して前記モバイルノードが新しいネットワーク接続ポイントに移動したかどうかを判定するステップと、
（ｃ）前記ステップ（ｂ）を受けて信号を生成するステップとを含むことを特徴とする方法。
【請求項２６】
請求項２５記載の方法であって、
ネットワーク・トポロジ・マップを保持するステップ、及び前記ネットワーク・トポロジ・マップを使用して前記ステップ（ｃ）を行うステップをさらに含むことを特徴とする方法。
【請求項２７】
請求項２５記載の方法であって、
前記ステップ（ｃ）はローミング信号を生成するステップを含むことを特徴とする方法。
【請求項２８】
請求項２５記載の方法であって、
前記ステップ（ｂ）はネットワーク・アダプタから前記ネットワーク接続ポイントの識別情報を取得するステップを含むことを特徴とする方法。
【請求項２９】
請求項２５記載の方法であって、
一般的な信号をサポートするネットワーク・インターフェイスが利用不可の場合、選択的ローミング検出メカニズムにフォールバックするステップをさらに含むことを特徴とする方法。
【請求項３０】
請求項２５記載の方法であって、
前記ネットワーク接続ポイント情報の少なくとも一部に応じて、代わりになるネットワーク接続パスの中から選択するステップをさらに含むことを特徴とする方法。
【請求項３１】
非接続のネットワークを介したモバイルシステムとの通信を円滑にするための方法であって、
第１ネットワークを介してノードと前記モバイルシステム間の通信を確立するステップと、
前記第１ネットワークと非接続の少なくとも第２ネットワーク上の前記ノードを識別するデータを、前記第１ネットワークを介して前記モバイルシステムに送信するステップと、
前記第２ネットワークを介して前記モバイルシステムと前記ノードとの通信を確立するために前記データを使用するステップとを含むことを特徴とする方法。
【請求項３２】
請求項３１記載の方法であって、
前記第１ネットワークを介して前記ノードに前記データを送信する前に、前記第２ネットワークを介して前記ノードとの通信許可を付与するために前記モバイルシステムを認証するステップをさらに含むことを特徴とする請求項３１記載の方法。
【請求項３３】
請求項２１記載の方法であって、
前記送信ステップは、分配されるインターフェイスデータを前記第１ネットワークを介して前記モバイルシステムに送信するステップを含むことを特徴とする方法。
【請求項３４】
前記モバイルコンピュータ装置のネットワーク・インターフェイス・アダプタは、前記ネットワークと物理的に接続されていることを特徴とする請求項１２記載のネットワーク。
【請求項３５】
前記モバイルコンピュータ装置はネットワーク接続ポイントと無線で通信することを特徴

とする請求項１２記載のネットワーク。
【請求項３６】
モバイルコンピュータシステムが複数の別個のネットワーク間を移動する際、モバイルコンピュータシステムとネットワークノードとの間での通信を維持するための方法であって、
第１ネットワークセグメントを介してモバイルシステムとノード間の通信を確立するステップと、
第１ネットワークセグメントを介して、それぞれ第１ネットワークセグメントとは別個の複数のさらなるネットワーク・セグメントを介して前記ノードとの通信を再確立する際に使用される情報をモバイルコンピュータシステムに送信するステップと、
前記情報を使用して、前記別個の複数のさらなるネットワーク・セグメントのいずれかを介してモバイルコンピュータシステムとノード間の通信を再確立するステップとを含むことを特徴とする方法。
【請求項３７】
請求項３６記載の方法であって、
前記情報は分配されるインターフェイスデータを含むことを特徴とする方法。
【請求項３８】
複数の別個のセグメントを有するネットワークにおいて最少コストルーティングを提供するための方法において、
（ａ）ネットワークと、一時的に接続されたモバイルコンピュータ装置との間の通信を確立するステップと、
（ｂ）一時的に接続されたモバイルコンピュータ装置が前記複数の別個のセグメント間を移動することができるローミング　構を使用するステップと、
（ｃ）モバイル・コンピューティングのローミングに応じて、ネットワークとモバイルコンピュータ装置との間の通信を再確立するための選択的で有効なネットワークパスを効率的に自動選択可能にするように、少なくとも１つのポリシーパラメータに実行させるステップとを含むことを特徴とする方法。
【請求項３９】
請求項３８記載の方法であって、
前記ポリシーパラメータは、帯域幅、１データ単位当りのコスト、およびサービス品質からなるグループの中から選択された要素を含むことを特徴とする方法。
【請求項４０】
少なくとも１つのピア・コンピュータシステムと、物理的リンクを介してネットワークに接続された少なくとも一つのモバイルコンピュータ装置とを含むモバイル・コンピューティング・ネットワークにおいて、
ネットワークに接続されたサーバを含み、前記サーバは、モバイルコンピュータ装置への物理的リンクが一時的に中断される間に、モバイルコンピュータ装置とピア・コンピュータシステムとの間の継続的な仮想データストリーム接続を維持するために、モバイルコンピュータ装置とピア・コンピュータシステムとの間の通信をプロキシすることを特徴とするモバイル・コンピューティング・ネットワーク。
【請求項４１】
前記モバイルコンピュータ装置は前記ネットワーク上の位置アドレスを有し、前記ピア・コンピュータシステムは仮想アドレスを使用して前記サーバと通信し、前記サーバは前記仮想アドレスを前記位置アドレスと対応付けることを特徴とする請求項４０記載のネットワーク。
【請求項４２】
前記サーバは、前記モバイルコンピュータ装置がその位置アドレスをいつ変更したかを検出し、前記仮想アドレスを前記変更された位置アドレスと再び対応付けることを特徴とする請求項４１記載のネットワーク。
【請求項４３】

前記サーバは、前記モバイルコンピュータ装置が一時的に圏外に出たり、またはローミング中である時に、前記モバイルコンピュータ装置に代わって、キューに入れ、前記ピア・コンピュータシステムからの要求に応答することを特徴とする請求項４０記載のネットワーク。

【請求項４４】

前記サーバは、従来のトランスポート・プロトコルを用いて前記モバイルコンピュータ装置と通信することを特徴とする請求項４０記載のネットワーク。

【請求項４５】

前記サーバは、遠隔プロシージャ・コールを用いて前記モバイルコンピュータ装置と通信することを特徴とする請求項４４記載のネットワーク。

【請求項４６】

前記サーバは、インターネット・モビリティ・プロトコルを用いて前記モバイルコンピュータ装置と通信することを特徴とする請求項４４記載のネットワーク。

【請求項４７】

前記インターネット・モビリティ・プロトコルは、ユーザ設定可能なタイムアウトに基づいてデータグラムの自動除去を行うことを特徴とする請求項４６記載のネットワーク。

【請求項４８】

前記インターネット・モビリティ・プロトコルは、ユーザ設定可能な再試行に基づいてデータグラムの自動除去を行うことを特徴とする請求項４６記載のネットワーク。

【請求項４９】

前記サーバは、前記モバイルコンピュータ装置によるネットワーク・リソースの消費に関するユーザ毎のポリシー管理を行うことを特徴とする請求項４０記載のネットワーク。

【請求項５０】

前記サーバは、ユーザ設定可能なセッション優先順位を、前記モバイルコンピュータ装置の前記セッションに付与することを特徴とする請求項４０記載のネットワーク。

【請求項５１】

前記モバイルネットワークは複数のサブネットワークを含み、前記モバイルコンピュータ装置は、前記モバイルコンピュータ装置によって前記複数のサブネットワークのうちの１つから前記複数のサブネットワークの別の１つへのローミングを可能にする他の手順と共に動的ホスト構成プロトコルを用いることを特徴とする請求項４０記載のネットワーク。

【請求項５２】

前記サーバは、モビリティ管理サーバを含むことを特徴とする請求項４０記載のネットワーク。

【請求項５３】

前記モバイルコンピュータ装置と前記サーバとを接続する少なくとも１つのモバイル相互接続をさらに含むことを特徴とする請求項４０記載のネットワーク。

【請求項５４】

モバイルコンピュータ環境において少なくとも一つのモバイルコンピュータ装置との持続的な接続を維持する方法であって、

前記モバイルコンピュータ装置と、少なくとも一つのさらなるコンピュータ装置との間の少なくとも１つのセッションを管理するステップと、

モバイルコンピュータ装置が圏外に出たり、サスペンドしたり、あるいはネットワークアドレスを変更した時に、セッションを維持するステップとを含むことを特徴とする方法。

【請求項５５】

請求項５４記載の方法であって、

前記セッションのために少なくとも１つのユーザ設定可能なセッション優先順位を付与するステップをさらに含むことを特徴とする方法。

【請求項５６】

請求項５４記載の方法であって、

前記管理ステップは、前記モバイルコンピュータ装置によるネットワーク・リソースの消

費を管理するステップを含むことを特徴とする方法。
【請求項５７】
請求項５４記載の方法であって、
前記モバイルコンピュータ環境は複数のサブネットワークを含み、前記維持ステップでは、動的ホスト構成プロトコルを使用して前記モバイルコンピュータ装置が前記サブネットワーク間を移動する時にセッションを維持することを特徴とする方法。
【請求項５８】
請求項５４記載の方法であって、
前記管理ステップは、前記モバイルコンピュータ装置とデータグラムのやり取りを行い、少なくとも１つのユーザ設定可能なパラメータに基づき前記データグラムのうち信頼性の低いデータグラムを自動的に除去することを特徴とする方法。
【請求項５９】
請求項５８記載の方法であって、
前記ユーザ設定可能なパラメータは、タイムアウトを含むことを特徴とする方法。
【請求項６０】
請求項５８記載の方法であって、
前記ユーザ設定可能なパラメータは、ユーザ設定可能な再試行番号を含むことを特徴とする方法。
【請求項６１】
請求項５４記載の方法であって、
前記モバイルコンピュータ装置に可変の位置アドレスを付与するステップをさらに含み、前記管理ステップはセッションと関連付けされる仮想アドレスに前記位置アドレスを対応付けするステップを含むことを特徴とする方法。
【請求項６２】
請求項５４記載の方法であって、
前記管理ステップは、遠隔プロシージャ・コール・プロトコルを用いてモバイルコンピュータ装置と通信するステップを含むことを特徴とする方法。
【請求項６３】
請求項５４記載の方法であって、
前記維持ステップは、前記モバイルコンピュータ装置を前記モバイルコンピュータ環境に接続する物理的リンクの中断時に前記セッションの接続状態を維持することを特徴とする方法。
【請求項６４】
請求項５４記載の方法であって、
前記管理ステップは、少なくとも１つの標準的なトランスポート・プロトコルを用いて前記モバイルコンピュータ装置と通信するステップを含むことを特徴とする方法。
【請求項６５】
請求項５４記載の方法であって、
前記モバイルコンピュータ装置は複数のアプリケーションソースを含み、前記管理ステップは、前記複数のアプリケーションソースからのデータをストリームに融合するステップと、前記ストリームを送信するステップとを含むことを特徴とする方法。
【請求項６６】
請求項６５記載の方法であって、
前記ストリームから融合されたデータを逆多重化して、前記逆多重化されたデータを複数の関連する目的地に送信するステップをさらに含むことを特徴とする方法。
【請求項６７】
請求項６５記載の方法であって、
前記ストリームはフレームを含み、前記融合処理は、モバイルコンピュータ環境の最大伝送単位に適合するように、前記フレームに対して動的なサイズ変更を行う処理を含むことを特徴とする方法。

【請求項６８】
請求項６５記載の方法であって、
前記融合処理は、信頼性の低いデータのセマンティクスを維持する処理と、前記信頼性の低いデータを前記セマンティクスに基づき選択的に破棄する処理とを含むことを特徴とする方法。
【請求項６９】
請求項５４記載の方法であって、
前記管理ステップは、前記モバイルコンピュータ装置へのメッセージの保証配信、および／または前記モバイルコンピュータ装置からのメッセージの保証配信を行う処理を含むことを特徴とする方法。
【請求項７０】
請求項５４記載の方法であって、
前記管理ステップは、前記モバイルコンピュータ装置がどのネットワーク・リソースにアクセス可能かを制御する処理を含むことを特徴とする方法。
【請求項７１】
少なくとも一つのさらなるコンピュータ装置を含むモバイルコンピュータ環境において少なくとも一つのモバイルコンピュータ装置との持続的な接続を維持するためのサーバであって、
モバイルコンピュータ装置が圏外に出たり、サスペンドしたり、あるいはネットワークアドレスを変更した時にセッションを維持し、前記モバイルコンピュータ装置と少なくとも一つのさらなるコンピュータ装置間の少なくとも１つのセッションを管理するセッション・マネージャを含むサーバ。
【請求項７２】
前記セッション・マネージャは、前記セッションのためにユーザ設定可能なセッション優先順位を少なくとも１つ付与するセッション優先キューを含むことを特徴とする請求項７１記載のサーバ。
【請求項７３】
前記セッション・マネージャは、前記モバイルコンピュータ装置によるネットワーク・リソースの消費を管理する手段を含むことを特徴とする請求項７１記載のサーバ。
【請求項７４】
前記モバイルコンピュータ環境は、複数のサブネットワークを含み、前記セッション・マネージャは、動的ホスト構成プロトコルを使用して前記モバイルコンピュータ装置が前記サブネットワーク間を移動する時にセッションを維持することを特徴とする請求項７１記載のサーバ。
【請求項７５】
前記セッション・マネージャは、前記モバイルコンピュータ装置とデータグラムのやり取りを行い、少なくとも１つのユーザ設定可能なパラメータに基づき前記データグラムのうち信頼性の低いデータグラムを自動的に除去することを特徴とする請求項７１記載のサーバ。
【請求項７６】
前記ユーザ設定可能なパラメータは、タイムアウトを含むことを特徴とする請求項７５記載のサーバ。
【請求項７７】
前記ユーザ設定可能なパラメータは、ユーザ設定可能な再試行番号を含むことを特徴とする請求項７５記載のサーバ。
【請求項７８】
前記モバイルコンピュータ環境は、前記モバイルコンピュータ装置に可変の位置アドレスを付与し、前記セッション・マネージャはセッションと関連付けされる仮想アドレスに前記位置アドレスを対応付けすることを特徴とする請求項７１記載のサーバ。
【請求項７９】

前記セッション・マネージャは、遠隔プロシージャ・コール・プロトコルを用いてモバイルコンピュータ装置と通信することを特徴とする請求項７１記載のサーバ。
【請求項８０】
前記モバイルコンピュータ環境は、前記モバイルコンピュータ装置を前記モバイルコンピュータ環境に接続する少なくとも１つの物理的リンクを含み、前記セッション・マネージャは、前記物理的リンクの中断時に前記セッションの接続状態を維持することを特徴とする請求項７１記載のサーバ。
【請求項８１】
前記セッション・マネージャは、少なくとも１つの標準的なトランスポート・プロトコルを用いて前記モバイルコンピュータ装置と通信することを特徴とする請求項７１記載のサーバ。
【請求項８２】
前記モバイルコンピュータ装置は複数のアプリケーションソースを含み、前記セッション・マネージャは、前記複数のアプリケーションソースに関連するデータをストリームに融合して、前記ストリームを送信することを特徴とする請求項７１記載のサーバ。
【請求項８３】
前記モバイルコンピュータ装置は複数のアプリケーションソースを含み、前記セッション・マネージャは、前記複数のアプリケーションソースからの融合されたデータを逆多重化して、前記逆多重化されたデータを複数の関連する目的地に送信することを特徴とする請求項７１記載のサーバ。
【請求項８４】
前記セッション・マネージャは、フレームを用いて前記モバイルコンピュータ装置と通信し、モバイルコンピュータ環境の最大伝送単位に適合するように、前記フレームに対して動的なサイズ変更を行うことを特徴とする請求項７１記載のサーバ。
【請求項８５】
前記セッション・マネージャは、信頼性の低いデータのセマンティクスを維持し、前記信頼性の低いデータを前記セマンティクスに基づき選択的に破棄することを特徴とする請求項７１記載のサーバ。
【請求項８６】
前記セッション・マネージャは、前記モバイルコンピュータ装置へのメッセージの保証配信、および／または前記モバイルコンピュータ装置からのメッセージの保証配信を行うことを特徴とする請求項７１記載のサーバ。
【請求項８７】
前記セッション・マネージャは、前記モバイルコンピュータ装置がアクセス可能なネットワーク・リソースの制御を行うことを特徴とする請求項７１記載のサーバ。
【請求項８８】
プロキシ・サーバを含むモバイルコンピュータ環境において、モバイルコンピュータ装置が圏外に出たり、サスペンドしたり、あるいはネットワークアドレスを変更した時に、少なくとも一つのさらなるコンピュータ装置と持続的な仮想接続を維持するモバイルコンピュータ装置であって、
トランスポート・ドライバ・インターフェイスと、
前記トランスポート・ドライバ・インターフェイスと接続されるモバイル・インターセプタを含み、
前記モバイル・インターセプタは、前記トランスポート・ドライバ・インターフェイスでのネットワークサービス要求を傍受し、前記ネットワークサービス要求に応じて、遠隔プロシージャ・コールを生成し、前記遠隔プロシージャ・コールを前記プロキシ・サーバに送信することを特徴とするモバイルコンピュータ装置。
【請求項８９】
前記モバイル・インターセプタは、ユーザ設定可能なセッション優先順位を少なくとも１つ付与するセッション優先キューを含むことを特徴とする請求項８８記載のモバイルコン

ピュータ装置。
【請求項９０】
前記モバイル・インターセプタは、前記モバイルコンピュータ装置によるネットワーク・リソースの消費を管理する手段を含むことを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９１】
前記モバイルコンピュータ環境は、複数のサブネットワークを含み、前記モバイルコンピュータ装置は、前記モバイルコンピュータ装置が前記サブネットワーク間を移動する時に位置アドレスを取得するために動的ホスト構成プロトコルを使用する手段をさらに含むことを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９２】
前記モバイル・インターセプタは、前記プロキシ・サーバとデータグラムのやり取りを行い、少なくとも１つのユーザ設定可能なパラメータに基づき前記データグラムのうち信頼性の低いデータグラムを自動的に除去することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９３】
前記ユーザ設定可能なパラメータは、タイムアウトを含むことを特徴とする請求項９２記載のモバイルコンピュータ装置。
【請求項９４】
前記ユーザ設定可能なパラメータは、ユーザ設定可能な再試行番号を含むことを特徴とする請求項９２記載のモバイルコンピュータ装置。
【請求項９５】
前記モバイルコンピュータ装置は、前記モビリティ管理サーバを仮想アドレスと対応付ける、関連付けされた可変の位置アドレスを有することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９６】
前記モバイル・インターセプタは、遠隔プロシージャ・コール・プロトコルを用いて前記モビリティ管理サーバと通信することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９７】
前記モバイルコンピュータ環境は、前記モバイルコンピュータ装置を前記モバイルコンピュータ環境に接続する少なくとも１つの物理的リンクを含み、前記モバイル・インターセプタは、前記物理的リンクでの中断後に前記モビリティ管理サーバから少なくとも１つのセッションの更新された接続状態情報を受け取ることを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９８】
前記モバイルコンピュータ装置は標準的なトランスポート・プロトコル・ハンドラを含み、前記モバイル・インターセプタは前記標準的なトランスポート・プロトコル・ハンドラを介して前記モビリティ管理サーバと通信することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項９９】
前記モバイルコンピュータ装置は複数のアプリケーションソースを含み、前記モバイル・インターセプタは前記複数のアプリケーションソースに関連するデータをストリームに融合して、前記ストリームを前記モビリティ管理サーバに送信することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項１００】
前記モバイルコンピュータ装置は複数のアプリケーション目的地を含み、前記モバイル・インターセプタは、前記複数のアプリケーションソースからの融合されたデータを逆多重化して、前記逆多重化されたデータを複数のアプリケーション目的地に送信することを特徴とする請求項８８記載のモバイルコンピュータ装置。

【請求項１０１】
前記モバイル・インターセプタは、フレームを用いて前記プロキシ・サーバと通信し、モバイルコンピュータ環境の最大伝送単位に適合するように、前記フレームに対して動的なサイズ変更を行うことを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項１０２】
前記モバイル・インターセプタは、信頼性の低いデータのセマンティクスを維持し、前記信頼性の低いデータを前記セマンティクスに基づき選択的に破棄することを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項１０３】
前記モバイル・インターセプタは、前記プロキシ・サーバへのメッセージの保証配信、および／または前記プロキシ・サーバからのメッセージの保証配信を行うことを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項１０４】
前記モバイル・インターセプタは、前記モバイルコンピュータ装置がアクセス可能なモバイルコンピュータ環境のリソースの制御を行うことを特徴とする請求項８８記載のモバイルコンピュータ装置。
【請求項１０５】
トランスポート・ドライバ・インターフェイスと、
前記トランスポート・ドライバ・インターフェイスと接続されるモバイル・インターセプタを含む少なくとも一つのモバイルコンピュータ装置を含むモバイルコンピュータ環境であって、
前記モバイル・インターセプタは、前記トランスポート・ドライバ・インターフェイスでのネットワークサービス要求を傍受し、前記ネットワークサービス要求を受けて、遠隔プロシージャ・コールを生成し、前記遠隔プロシージャ・コールを少なくとも１つのプロキシ・サーバに送信し、
前記プロキシ・サーバは、前記モバイル・インターセプタによって送信された前記遠隔プロシージャ・コールを受信及び処理するワーク・ディスパッチャを少なくとも１つ含み、
前記プロキシ・サーバは、前記モバイルコンピュータ装置が前記モバイルコンピュータ環境と一時的に切断された時に、前記モバイルコンピュータ装置の代わりに仮想セッションをプロキシするプロキシ・キューを含むことを特徴とするモバイルコンピュータ環境。
【請求項１０６】
ネットワーク接続ポイントを介してネットワークと接続されるモバイルコンピュータ装置を少なくとも一つ含むモバイル・コンピューティング・ネットワークにおいて、ネットワーク接続ポイントの識別子の少なくとも一部分に基づき、前記モバイルコンピュータ装置が異なるネットワーク・セグメントに移動したかどうか検出するインターフェイス補助ローミングリスナを含むことを特徴とするモバイル・コンピューティング・ネットワーク。
【請求項１０７】
前記モバイルコンピュータ装置はネットワーク・インターフェイス・アダプタを含み、前記リスナは前記ネットワーク・インターフェイス・アダプタから前記ネットワーク接続ポイントの識別子を取得することを特徴とする請求項１０６記載のネットワーク。
【請求項１０８】
前記リスナは、前記ネットワーク接続ポイントの識別子をネットワーク接続についての詳細情報に相関させる情報を記憶するネットワーク・トポロジ・マップを保持することを特徴とする請求項１０６記載のネットワーク。
【請求項１０９】
前記リスナは、前記ネットワークとの通信がいつ中断または再確立されるかを検出することを特徴とする請求項１０６記載のネットワーク。
【請求項１１０】
前記インターフェイス補助ローミングリスナは、（ａ）ネットワーク通信の中断及び再確立、及び（ｂ）前記ネットワーク接続ポイントの識別子の変化の検出を受けて、ローミン

グ信号を生成することを特徴とする請求項１０９記載のネットワーク。
【請求項１１１】
前記モバイルコンピュータ装置において使用されるインターフェイスベースのリスナであって、インターフェイスベースのリスナは少なくとも１つのネットワーク・インターフェイス・アダプタからの情報と、少なくとも１つのネットワークスタックから入手可能な情報を統合して、前記モバイルコンピュータ装置が新しいネットワーク接続ポイントに移動したかどうか判定することを特徴とするインターフェイスベースのリスナ。
【請求項１１２】
ネットワーク接続ポイント情報を含むネットワーク接続情報を提供するネットワーク・トポロジ・マップを含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１３】
上記リスナは学習情報に基づき前記ネットワーク・トポロジ・マップを動的に構成することを特徴とする請求項１１２記載のリスナ。
【請求項１１４】
イベント発生に基づき、ステータスをチェックするステータスチェッカーを含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１５】
前記イベントはタイマーのタイムアウト、ローレベルのローミング・ドライバ・コールバック、ネットワーク・レベル・アクティビティ・ヒントのいずれかを含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１６】
モバイルコンピュータ装置が既に現在のネットワーク接続ポイントを訪れたかどうかインターフェイスに照会する接続情報探索部を含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１７】
新しいネットワーク・セグメントにおいて有効となる現行のアドレスを登録または再取得する接続構成を含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１８】
インターフェイスから提供された情報の少なくとも一部分に基づいて、前記モバイルコンピュータ装置が異なるネットワーク・セグメントに移動したことの検出を受けてローミング信号を生成するローミング信号発生器を含むことを特徴とする請求項１１１記載のインターフェイスベースのリスナ。
【請求項１１９】
予め割り当てられたアドレスがまだ有効かどうかを判定するヒューリスティック・アナライザをさらに含むことを特徴とする請求項１１８記載のインターフェイスベースのリスナ。
【請求項１２０】
モバイルノードが新しいネットワーク接続ポイントに移動したかどうか判定する方法であって、
（ａ）ネットワーク・インターフェイスからネットワーク接続ポイントの識別情報を受け取るステップと、
（ｂ）前記ネットワーク接続ポイントの識別情報を使用して前記モバイルノードが新しいネットワーク接続ポイントに移動したかどうかを判定するステップと、
（ｃ）前記ステップ（ｂ）を受けて信号を生成するステップを含むことを特徴とする方法。
【請求項１２１】
請求項１２０記載の方法であって、
ネットワーク・トポロジ・マップを保持するステップ、及び前記ネットワーク・トポロジ

・マップを使用して前記ステップ（ｃ）を行うステップをさらに含むことを特徴とする方法。
【請求項１２２】
請求項１２０記載の方法であって、
前記ステップ（ｃ）はローミング信号を生成するステップを含むことを特徴とする方法。
【請求項１２３】
請求項１２０記載の方法であって、
前記ステップ（ｂ）はネットワーク・アダプタから前記ネットワーク接続ポイントの識別情報を取得するステップを含むことを特徴とする方法。
【請求項１２４】
請求項１２０記載の方法であって、
一般的な信号をサポートするネットワーク・インターフェイスが利用不可の場合、選択的ローミング検出機構にフォールバックするステップをさらに含むことを特徴とする方法。
【請求項１２５】
請求項１２０記載の方法であって、
前記ネットワーク接続ポイント情報の少なくとも一部分を受けて、択一的なネットワーク接続パスから選択するステップをさらに含むことを特徴とする方法。
【発明の詳細な説明】
［発明の背景］
本発明は、ネットワークによって接続されたコンピュータ装置間の接続性に関する。より詳細には、ノマディック（ｎｏｍａｄｉｃ）システムの特性を透過性（トランスペアレント）を有する状態で扱うとともに、既存のネットワーク・アプリケーションを、対応するモバイル環境で確実に動作させる方法とシステムに関する。さらに詳細には、断続的に接続される、携帯型データ端末（ｈａｎｄｈｅｌｄ　ｄａｔａ　ｕｎｉｔｓ）やパソコン装置などの装置間における、連続的なデータストリームのやり取りを可能にする技術およびシステムに関する。
【０００１】
［発明の背景および要約］
近年、各企業は、重要な情報への迅速なアクセスこそが競争優位を保つための方策であると認識するようになってきた。このことから、特に安価なラップトップやハンドヘルドのコンピュータ装置の普及とも相まって、モバイルなどの断続的な接続が行われるコンピュータ装置が急速に企業ネットワークの重要な一要素となりつつある。しかしながら、こうしたノマディックな装置を既存のネットワーク・インフラへと統合することは、各企業の情報管理担当者にとって頭の痛い問題ともなっている。
【０００２】
モバイルネットワーキングに関する問題の多くは、イーサネット（登録商標）登場以前にローカルエリア・ネットワーク（ＬＡＮ）構築に伴っていた問題に近い。即ち、モバイル・プロトコルやインターフェイスには多くの種類があり、また、まだ規格が制定段階にあることから、異なるシステム間の相互運用性は無いも同然である。さらに、上記のようなネットワーク技術による通信はたいてい低速で帯域幅も限られており、各システムの特殊性から、アップデートに伴う費用も高額になっている。
【０００３】
こうした問題に加えて、モバイル技術は、以下のような固有の問題も有している。即ち、メインのネットワークと相互接続するの際には公共のネットワーク・インフラを経由する場合があるが、この際に機密情報が傍受されてしまう危険性がある。さらに、無線装置を経由して相互接続する場合であれば、機密情報がいわば「放送」されてしまうことになり、類似のインターフェイスを所有する誰もが該情報をたいした障害もなく傍受することができてしまう。
【０００４】
しかし、おそらく上記のような問題よりもさらに重要な問題としては、従来、モバイルネ

ットワーキングが使われるのは基本的にメッセージ指向あるいはステートレスなアプリケーションに限られていて、クライアント／サーバ、ホスト／ターミナル方式をとるウェブベースあるいはファイル共有型のシステムモデルを利用した、既存あるいは新規の業務用アプリケーションには適さなかったことである。これは、一般的な業務用アプリケーションが効果的かつ確実に動作するためには、ステートレスなパケット交換だけでなく、連続的なデータストリームを用いたステートフルなセッションが必要であるという理由からであった。

【０００５】

このため、主要な市販のネットワーク・アプリケーションのほとんどは、ＴＣＰ／ＩＰセッションか、専用の仮想回路を必要としている。このようなセッションは、ネットワークが中断された場合はそれ以上機能できず、また接続時には、ネットワーク間のローミング（ネットワークアドレスの変更）が不可能である。さらに、モバイルネットワーキングは本質的に動的であり信頼性が低い。これらの問題について、以下では、モバイルネットワークにおいてよく見られる状況をいくつか考察することにする。

【０００６】

（非接続あるいは圏外のユーザ）

モバイル装置が所与のネットワークから切断されるかあるいはネットワークとのコンタクトを失う（つまり、無線相互接続の圏外に出たりネットワークがカバーしていない「穴」に入ったりする）と、携帯装置上で動作しているセッション指向のアプリケーションはピアとのステートフルな接続を失い、動作を停止する。装置が再び取り付けられるかあるいは装置とネットワークとのコンタクトが回復すると、ユーザはネットワークとの再接続を行い、セキュリティ確保のために再ログインを実行して、アプリケーションにおいてどこで作業が中断されたかを同定し、必要ならば失われたデータの再入力を行わなければならない。このような再接続処理は時間を要し、費用もかかり、またユーザを非常に苛立たせることにもなる。

【０００７】

（他のネットワークへの、あるいはルータ境界を越える移動（ネットワークアドレス変更））

モバイルネットワークは通常、管理容易性の見地からセグメント化されているが、一方でモバイル装置はその用途上、ローミングが可能になっている。相互接続されたネットワーク間のローミングとは、ネットワークアドレスが変更されるということを意味する。ネットワークアドレスがシステムの動作中に変更される場合、対応しているピア同士の接続を保つためにルーティング情報を変更することが必要になる。さらに、新しいネットワークアドレスを取得するために、それまでに確立されたステートフルなアプリケーションのセッション全てを終了させなければならない場合もあり、上記の再接続処理に関わる問題がここでも浮上する。

【０００８】

（セキュリティ）

既に述べたように、各企業は機密情報を守る必要がある。市販のアプリケーションの多くは、物理的ネットワークへのアクセスがコントロールされ（つまり、安全な施設の内部に構築されたケーブルを使って実行され）、セキュリティは、認証や場合によっては暗号化に関する付加的な層を通じて保持されることを前提として作成されている。しかしながら、ノマディック・コンピューティングではこうした前提は自明のものではない。というのも、公共の電波や有線インフラを通過する際にデータが傍受されてしまう危険性があるからである。

【０００９】

このため、ノマディックなシステムの特性を透過性を有するものとすることによって、既存のネットワーク・アプリケーションが種々のモバイル環境で確実に動作することを可能にした、統合的なソリューションの登場が大いに望まれる。

【００１０】

本発明は、上記の問題を解決するために、企業内ネットワークを延長して、ネットワーク管理者がモバイル装置のユーザも固定装置のユーザと同じアプリケーションに容易にアクセスできるようにすることを可能にすると同時に、信頼性やネットワーク管理の一元性も失わない、一貫したソリューションを提示する。該ソリューションは、これまでの有線ネットワーク規格の長所を、発展しつつあるモバイルネットワーキングに関する規格にも持たせることで、既存のネットワーク・アプリケーションにも対応したものとなっている。

【００１１】

本発明の非限定的で例示的具体的な実施の形態の一側面においては、モバイル相互接続（ｍｏｂｉｌｅ ｉｎｔｅｒｃｏｎｎｅｃｔ）に接続されたモビリティ管理サーバ（Ｍｏｂｉｌｉｔｙ Ｍａｎａｇｅｍｅｎｔ Ｓｅｒｖｅｒ；ＭＭＳ）は、数量を限定されないモバイル端末システム（Ｍｏｂｉｌｅ Ｅｎｄ Ｓｙｓｔｅｍｓ；ＭＥＳ）それぞれの状態の維持や、ネットワークへおよびピアにおけるアプリケーション処理への持続的な接続の維持に必要な、複雑なセッションの管理を行う。あるモバイル端末システムが、圏外に出てしまったり、サスペンドしたり、あるいは（あるモバイル相互接続から別の相互接続へとローミングすることなどで）ネットワークアドレスを変更したりした場合でも、モビリティ管理サーバは該システムと、該システムと対応しているピアとの接続を維持する。言い換えれば、該モバイル端末システムは、実際には物理的な接続が一時中断されてしまうにも関わらず、ピアとの仮想的な接続を維持し続けることになる。

【００１２】

また、上記本発明の非限定的で具体的な実施の形態は、以下の（これらに限定されないが）新規かつ有用な技術、構成を提示する。

【００１３】

・ユーザによって設定可能なセッション特性をモバイル・クライアントに付与するモビリティ管理サーバ。

【００１４】

・ネットワーク・リソースの消費に関し、ユーザごとにモバイル装置のポリシーを管理。

【００１５】

・工業規格である動的ホスト構成プロトコル（Ｄｙｎａｍｉｃ Ｈｏｓｔ Ｃｏｎｆｉｇｕｒａｔｉｏｎ Ｐｒｏｔｏｃｏｌ；ＤＨＣＰ）をモビリティ管理サーバと連携させて用いるローミング方法。

【００１６】

・ユーザによって設定可能なタイムアウト条件に基づいて、信頼性の低いデータグラムを自動的にシステムから除去。

【００１７】

・ユーザによって設定可能な再試行条件に基づいて、信頼性の低いデータグラムを自動的にシステムから除去。

【００１８】

より詳細には、上記本発明の好ましい具体的な実施形態の一例において、モバイル相互接続ネットワークに接続されたモビリティ管理サーバが備えられている。該モビリティ管理サーバは、数量を限定されないモバイル端末システムそれぞれの状態を維持し、ネットワークへおよび他の処理（例えば他のネットワークベースのピア・システムにて実行される処理）への持続的な接続の維持に必要な、複合的なセッションを管理する。あるモバイル端末システムが、圏外に出てしまったり、サスペンドしたり、あるいは（あるモバイル相互接続から別の相互接続へとローミングすることなどで）ネットワークアドレスを変更したりした場合、モビリティ管理サーバはデータの受信と待　要求を認識して、該システムと、該システムと対応しているピアとの接続を維持する。このようにモビリティ管理サーバがプロキシとして働くことにより、モバイル端末システム上で動作するアプリケーションは、あるネットワーク媒体との物理的な接続が一時的に中断された場合でも、持続的な接続を維持することができる。

【００１９】

また、上記本発明の好ましい実施の形態の他の一例において、モビリティ管理サーバはモバイル端末システム用のアドレスを管理する。モバイル端末システムには、それぞれにプライマリネットワーク上のプロキシ・アドレスが割り当てられている。この非常に汎用性の高いアドレスはモバイル端末システムの「仮想アドレス」と呼ばれる。モビリティ管理サーバはこの仮想アドレスを、ノマディック・システムにおける現在の「位置」アドレスと対応付ける。モバイル端末システムの位置アドレスは、該システムがネットワーク間を移動する際に変更されるが、仮想アドレスの方は、どの接続がアクティブになっていようが、接続時間が長くなっていようが、上記アドレスが静的に割り当てられている限りは一定となる。

【００２０】

上記本発明の好ましい実施形態のさらに他の一例において、モビリティ管理サーバは、コンソール（制御）アプリケーションと包括的なメトリクスとによってモバイル端末システムの集中的な管理を実現する。さらに、好ましい実施形態では、ユーザによって設定可能なセッション特性を、プロキシ・サーバ上で実行されるモバイル・クライアントについて実現し、ネットワーク・リソースの消費に関し、ユーザごとにモバイル装置のポリシー設定を管理する。

【００２１】

さらに、上記一側面においては、遠隔プロシージャ・コール（ＲＰＣ）・プロトコル（Ｒｅｍｏｔｅ　Ｐｒｏｃｅｄｕｒｅ　Ｃａｌｌ　ｐｒｏｔｏｃｏｌ）およびインターネット・モビリティ・プロトコル（Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　Ｐｒｏｔｏｃｏｌ）が、プロキシ・サーバと各モバイル端末システムとの間の通信の確立に使用されている。

【００２２】

遠隔プロシージャ・コールによって、ローカルなシステムから、遠隔のシステムにおけるプロシージャを呼び出す処理が可能になり、ＲＰＣプロトコルを使用することで、モバイル端末システムが接続を切ったり、圏外に行ったり、動作をサスペンドしたりということを、アクティブなネットワークセッションを中断することなくできるようになる。このように、セッションの維持は専用のアプリケーションによってなされるわけではないので、市販のアプリケーションを、何ら変更することなくノマディックな環境下で使用することができるようになる。

【００２３】

ＲＰＣプロトコルは、トランザクションを、標準的なネットワークトランスポート・プロトコルおよびインフラを経由して送られるメッセージに生成する。このＲＰＣメッセージは、モバイル端末システムにおいて実行されているアプリケーションによって開始されたネットワーク・トランザクション全体を含んでおり、これにより、モビリティ管理サーバとモバイル端末システムとの間の接続状態情報を、両者の間の物理的な接続が途切れているときをも含め、常にシンクロさせておくことが可能になる。ＲＰＣを備える上記本発明の実施の形態において、プロキシ・サーバとモバイル端末システムとは、全ての時間における全ての共有接続に関する統一された論理データベースを管理すべく、各トランザクションの状態について十分な情報を共有している。

【００２４】

上記本発明の好ましい実施形態におけるインターネット・モビリティ・プロトコルは、有線のローカルエリア・ネットワークと、それよりも信頼性の低い、無線ＬＡＮおよびＷＡＮといったネットワークとの間の相違を補償する。フレームサイズとプロトコルタイミングとが調整されることで、モバイル環境を考慮していない通信に比べるとパフォーマンスが大幅に改善され、ネットワークのトラフィックは大きく減少する。これは、帯域幅が限られているときやバッテリーの寿命を考慮に入れなければならないときに、特に重要になる。さらに、上記本発明の好ましい実施形態におけるインターネット・モビリティ・プロトコルによって、公共のネットワークもしくは無線を通じて、モバイル端末システム―モビリティ管理サーバ間で伝送が行われる際の、機密データの安全性が確保される。インターネット・モビリティ・プロトコルは、認証された装置からのみプライベートなネットワ

ークにアクセスできるようにすることで、基本的なファイアウォールとしての機能も果たす。また、インターネット・モビリティ・プロトコルにより、モバイル端末システム―モビリティ管理サーバ間の全ての通信を認証、暗号化することも可能である。

【００２５】

上記本発明の好ましい実施形態のさらに他の一例において、モバイル相互接続は、標準的なネットワーク・アプリケーションのインターフェイスが適用できる範囲を広げるべく、標準的なトランスポート・プロトコル（例としてＴＣＰ／ＩＰ、ＵＤＰ／ＩＰ、ＤＨＣＰなど）を使用して構築される。上記本発明の好ましい実施形態によって、データ転送、セキュリティ、アドレス管理、装置管理、ユーザ管理が効果的に統合され、ノマディック環境を効果的に透過性を有する状態とすることができる。インターネット・モビリティ・プロトコルは、複数のデータストリーム（信頼度の高いものも低いものも）を、標準的なネットワーク・インフラ上で標準的なトランスポート・プロトコルによって与えられる、単一の仮想チャネルを通じて多重化するための効果的な方法を提供する。

【００２６】

ＲＰＣ層を用いて、インターネット・モビリティ・プロトコルは、違うソースから供給され、違うまたは同じ目的地へと向かうデータを融合させて単一のデータストリームとし、これをモバイルリンクを介して送る。該モバイルリンクの反対側の端において、該データストリームは逆多重化されて複数の異なったデータストリームとなり、それぞれの最終的な目的地に伝送される。この多重化／逆多重化により、利用可能な帯域幅を（最大限のサイズのネットワークフレームを生成することで）最大限に使うことができ、複数のチャネルを確立することができる（よって、優先順位付けが可能になり、基礎をなしているネットワークがデータ通信サービスを供給している場合であれば、その品質を保証することができる可能性もある）。

【００２７】

上記本発明の実施の形態に関するインターネット・モビリティ・プロトコルは、さらに以下のような特徴や利点を実現する。なお、本発明は以下の点に限定されるものではない。

【００２８】

・トランスポート・プロトコルの独立性

・ネットワーク上における位置（ｐｏｉｎｔ　ｏｆ　ｐｒｅｓｅｎｃｅ；　ＰＯＰ）あるいはネットワーク・インフラを、データの流れに影響を与えることなく変更可能（物理的境界、ポリシー、あるいは帯域幅による制約が無い場合のみ）

・付加的な経費が最小限

・伝送媒体に適した自動的なフラグメントのサイズ変更（あるフレームのプロトコル・データ・ユニットがネットワーク媒体の利用可能な上限の伝送ユニットよりも多いとき、インターネット・モビリティ・プロトコルは該フレームをフラグメント化し、再構築する。このことにより、該フレームのネットワーク伝送を保障することが可能となる。再伝送の際、該フレームは再度検査される。ネットワーク・インフラもしくは環境が変化していた場合、該フレームは再度フラグメント化されるか、あるいは伝送ユニットの上限が実際に上昇しているときには単一のフレームとして伝送される。）

・再伝送の際に、フレームに信頼性の低いデータを破棄させることで、信頼性の低いデータのセマンティクスを保持

・信頼性の高いデータグラムサービスにおける新しいセマンティクスを提供（これにより、インターネット・モビリティ・プロトコルによるピアの端末へのデータグラムの伝達が保証され、要求しているエンティティに伝達の通知がなされる）

・上りと下りの伝送パスをそれぞれ別に扱って、自動的に操作パラメータを調整し最適なスループットを実現（ヒステリシスに基づいて、フレームサイズ／フラグメント化の閾値、待機中のフレーム数（ウィンドウ）、再伝送時間、およびネットワークを通じて送られる複製データの量を減少させるための遅延承認時間などのパラメータを調整）。

【００２９】

・ネットワーク障害に対する耐性（モバイル環境での使用が想定されているため、一時的

にネットワーク媒体間の接続が切断されても、仮想チャネルが切断されたりアプリケーションベースの接続が切断されたりしてしまうことがない）
・操作パラメータを調整するための帯域内信号方式をピアに対して提供（接続された端末のそれぞれから、そのピアに対してネットワーク・トポロジや環境の変化に関する警告を出すことが可能）
・輻輳回避アルゴリズムを採用し、必要なときにはスループットを効果的に減衰
・選択的な確認応答と高速での再伝送とによって無駄な再伝送の回数を制限し、ノマディック環境におけるハンドオフ回復のスピードアップを実現（これにより、プロトコルはロスの多いネットワーク環境においても最適なスループットを維持）
・複数のフレームを待機状態にする、スライディング・ウィンドウ技術を採用（このパラメータは各方向に調整可能で、ピアからの確認応答を要することなくフレームを所定の上限までストリーミングするために与えられる）
・シーケンス番号が非バイト指向であることにより、単一のシーケンス番号で最大のペイロード・サイズまでを表現可能
・セキュリティ対策（インターネット・モビリティ・プロトコル層に認証層と暗号化層を追加可能）
・圧縮によって、帯域幅の限られた接続における効率性を確保
・どちらのピアも新たな位置に移動することが可能な平衡型設計
・どちらの側からでもピアへの接続が確立可能
・休止している接続を迅速に破棄して消費されていたリソースを回復する、休止タイムアウトを発動可能
・接続に対して任意の最大持続時間を設定可能（例えば、所定の期間経過後あるいは所定の日時の後に、接続の終了および／または拒否が可能）。

【００３０】
本発明の好ましい具体的な実施形態においては、システム管理者によるネットワーク・リソースの消費の管理が可能である。例えば、システム管理者は、モバイル端末システム、モビリティ管理サーバの少なくとも一方をコントロールすることができる。この場合のコントロールとは、例えばネットワーク帯域幅や他のリソースの配分の管理や、セキュリティ関連の事項を指す。管理はクライアント側で多数のリソースを持っているクライアントについて実行するのが効果的である。しかしながら、リソースを多く持たないシン・クライアントにポリシー管理のための付加的なコードや処理を負わせるのは望ましくない。よって、シン・クライアントの管理についてはモビリティ管理サーバ等によって集中的に行う、あるいはその一部を分担するのが最も現実的な解決策であると考えられる。モビリティ管理サーバがモバイル端末システムの各データストリームをプロキシすることによって、ポリシー管理が集中的に行われる。さらに、モビリティ管理サーバがユーザごとにプロキシを行うので、ユーザごと、装置ごとに、ネットワーク・リソースへのアクセスを管理し制限することができる。

【００３１】
簡単な例を挙げると、モビリティ管理サーバは、特定のユーザによるあるネットワーク・リソースへのアクセスを「締め出す」（ｌｏｃｋ　ｏｕｔ）ことができる。この点は、インターフェイスのネットワークが、モバイル相互接続によって組織の管理下にある施設の境界よりも外に「延長」（ｅｘｔｅｎｄ）されてしまっていることを考えると非常に重要である（例えば、以前勤めていた職場のネットワークに、元従業員が外部からアクセスを試みるといった場合を考えよ）。これにとどまらず、モビリティ管理サーバによるポリシー管理はさらに多くの利点を提供しうる。例えば、モビリティ管理サーバによって、あるＵＲＬに特定のユーザのみがアクセスできるようにしたり、ネットワークサービスによる要求によって戻されるデータをフィルタリングしたり、ネットワークの帯域幅保全のためにデータを圧縮したりということが可能になるといった点である。このように、既存もしくは新規のアプリケーションレベルのサービスを、シームレスかつ透過的な形で強化することができる。

【００３２】
また、モバイル端末システムは「信頼性の低い」（ｕｎｔｒｕｓｔｅｄ）ネットワーク（つまり組織の管理が及ぶ範囲外のネットワーク）とも接続されるため、リモート接続中に悪質なサイバー攻撃に遭う可能性がある。モバイル端末システムとの間でポリシーやフィルタを共有することで、悪質な接続からのモバイル端末システムの保全、リモート・ノードにおける進入（ｉｎｇｒｅｓｓ）フィルタリング、そして企業インフラの集中管理のさらなるセキュリティ向上が可能になる。
【００３３】
本発明の実施形態の他の一例では、インターフェイスによって補助されたローミングのリスナ（ｌｉｓｔｅｎｅｒ）によって、モバイル端末システムが一般的な信号伝達をサポートしたインターフェイスを利用し、インターフェイスによって補助されたローミングを行うことが可能になる。本発明の上記実施形態の一例の一側面において、モバイル端末システムの、インターフェイスベースのリスナは、所定の事象（例えばコールバックや、タイマーによるタイムアウト、データの損失を示唆するネットワーク活動など）に際して、モバイル端末システムの媒体の接続状態が変化したか否かを判定する。これは例えば、リスナが、モバイル端末システムが切り離されてネットワークと通信できる状態でなくなったことを検知して、インターフェイスにこれを示唆するといったことを意味する。再接続の際、リスナは予め記録された接続識別情報（ａｔｔａｃｈｍｅｎｔ　ｉｄｅｎｔｉｆｉｃａｔｉｏｎ　ｉｎｆｏｒｍａｔｉｏｎ）におけるネットワークポイントを参照して、モバイル端末システムが同じネットワークポイントに再接続されたか否かを判定する。再接続が同じネットワークポイントになされていた場合、リスナはモバイル・クライアントに、トランスポート・レベルでの通信の再確立を進めることを警告する。再接続が別のネットワークポイントになされていた場合、リスナはモバイル端末システムが「ローミング」（ｒｏａｍ）状態であることを示し、現状におけるネットワーク・セグメントで使用可能なアドレスをシステムに取得させるようにする（これは例えば、現行のアドレスを新規のサブネットにおいて有効であるように登録することを含んでもよい）。リスナは（操作を介して学習した）ネットワーク・トポロジのマップを保持して、そのクライアントに対して生成される信号（「同一サブネット上のローミング」、「ローミング」等）の適否を判定する一助としてもよい。
【００３４】
上記本発明の非限定的な好ましい具体的実施形態の他の一例においては、モビリティ管理サーバ（ＭＭＳ）に「非接続ネットワーキング」（ｄｉｓｊｏｉｎｔ　ｎｅｔｗｏｒｋｉｎｇ）モードでアクセスすることが可能である。この新しいアルゴリズムによって、あるネットワーク・インフラからは別のネットワーク・インフラにおけるネットワークアドレスが分からないような非接続ネットワーク・トポロジにおいても、ＭＭＳとの通信を確立する／持続するのに使われる、代替のネットワークアドレスを動的／静的に見つけ出すことができるようになる。この構成により、ＭＭＳが利用可能な代替アドレスのリストが予め設定され、通信中にＭＥＳ（モバイル端末システム）に送られるかあるいはＭＥＳによって動的に学習される。実施の一形態において、ＭＭＳはＭＥＳに、一つ以上のＭＭＳネットワークアドレスもしくは他のネットワークに対応した他のＭＭＳのアイデンティティを、単一のネットワークによる通信によって送ることができる。該リストは、回路構築の際やその他接続が確立されている間のいかなる時にも、送付／更新が可能である。
【００３５】
ＭＥＳが別のネットワークへと移動するとき、ＭＥＳは該ネットワークにおける新規のネットワーク接続を通じてＭＭＳとコンタクトをとるために、ＭＭＳの「エイリアス」（ａｌｉａｓ）アドレス／アイデンティティのリストを用いる。これにより、移動前のネットワークと移動後のネットワークとがアドレス、ルートその他の情報を共有していなくても、ＭＥＳは新規のネットワーク接続を通じてＭＭＳとのコンタクトを再確立することができる。
【００３６】

上記本発明の実施の形態のさらに他の例において、ポリシー管理に関する意思決定は分散型モバイルネットワークトポロジの内部にて行われ、例えば、ルールベースのポリシー管理プロシージャが、様々なメトリクスに基づいて要求の遂行を許可、拒絶、または制限する。このような管理形態は、例えば、マルチホーム／パス環境における最少コストルーティングといったコストメトリクスに基づいて意思決定をする際に用いられる。

【００３７】

このようなポリシー管理技術では、意思決定の際にモビリティあるいは位置取得（つまりローミング）に関する事項が考慮に入れられてもよい。例えば、管理ルールが装置の位置（ネットワークのどの接続ポイント、例えばアクセスポイント／基地局、ハブ、ルータ、ＧＰＳ座標等に近いかなど）に基づいて決定されてもよく、これにより、特定の操作が、あるビル内では許可されるが別のビルでは許可されないといったことが可能になる。例えば複数の違った無線ネットワークを利用している企業の場合にこの構成が有用であると考えられる。このような企業において、例えば積込ドックとオフィスとが無線ネットワークで接続されている場合がある。システム管理者は、積込ドックにいる従業員（例えばユーザや装置）がオフィス環境の無線ネットワークにアクセスできないようにすることができる。さらに、ポリシー管理によって、許可、拒否、遅延という３つの状態のいずれかをもたらすようにすることも可能である（例えば、決定が帯域幅やコストに基づいてなされる場合、操作の実行はより適切な時期がくるまで遅延される）。

【００３８】

上記実施例におけるポリシー管理においては、決定に基づいて動作を変更することが可能となっている。ひとつのアクションの例としては、全てのアクティブなアプリケーションによる帯域幅の消費を抑えるというものがある。また、例えば、著しく帯域幅を消費する特定のアプリケーションが存在する場合に、ポリシーエンジン（ｐｏｌｉｃｙ ｅｎｇｉｎｅ）によって該アプリケーションによる操作／トランザクションの完了までの速度を管理することが可能となる。この動作は、誤ったアプリケーションの動作を抑制させることを動的に学習するようになっていてもよい。もうひとつのアクションの例として、データの復元（例えば、利用可能な／許可された帯域幅やコスト、ユーザによる制限に基づいたグラフィックイメージのディザリング）がある。

【００３９】

さらに、ルールエンジン（ｒｕｌｅｓ ｅｎｇｉｎｅ）は、ルール評価に基づいて他のアクションを発動させる。イベントをロギングする、警告を発する、ユーザにアクションが拒否、遅延、あるいは条件付けされたことを通知するといった外部プロシージャが実行されてもよい。これらのような通知は、既存のルールへのオーバーライドがオペレータから求められるというような、インタラクティブなものであってもよい。

【００４０】

上記実施例におけるポリシー管理エンジン（ｐｏｌｉｃｙ ｍａｎａｇｅｍｅｎｔ ｅｎｇｉｎｅ）において、その意思決定は、装置、装置のグループ、ユーザ・グループ、ユーザ、あるいはネットワーク接続ポイントに関連した、任意の数のメトリクスもしくはその組み合わせに基づいてなされていてもよい。

【００４１】

ポリシー管理機能の一部として、他にもローカルベースの情報やサービスが、ポリシー管理、ネットワーク・モデリング、および／またはアセット・トラッキングのために取得され／備えられていてもよい。このようなサービスには、モバイル端末システムの現在位置に関連した情報がユーザやモバイル端末システムに自動的に提供される機能が含まれる。該情報は、メッセージ、ファイルその他の電子的フォーマットにて供給されてもよい。

【００４２】

この機能の非限定的な利用例としては、ショッピングモール運営者、各種業界団体、大規模小売業者などがショッピングセンターに、ブルートゥース ＰＡＮ、ＩＥＥＥ ８０２．１１ ＬＡＮ、８０２．１５ ＰＡＮ、その他の無線ネットワーク規格に準拠した、無線アクセスポイントを戦略的に設けるというものがある。この場合、顧客がセンター内を

歩き回るのにあわせて、モバイル端末システムの現在位置周辺の店舗は顧客に情報を提示することができる。この情報としは、セールやディスカウント、特典などについてのものが含まれる。これに加え、モバイル端末システムに販促用の電子クーポン券が供給されるのもよい。店舗側は上記のようなサービスを、モール運営者、業界団体、小売業者、あるいはその他のサービス提供者による集中的な管理機構に登録することになる。

【００４３】

上記技術が利用される他の非限定的な例として、現場サービス、訪問販売、宅配などのバーティカル産業において、位置を基準にして情報を収集する、地域サービス、地図、方角、顧客、事故など公共の情報を、位置を基準にモバイル端末システムに送る、などがある。

【００４４】

さらに他の非限定的な例として、モバイル端末システムをモニタリング、トラッキングするウェブベースのサービスがある。例えば、顧客は該トラッキングサービスに登録し、信頼の置ける第三者機関が、ホスティングされているウェブサービスにログオンして顧客のモバイル端末システムの精確な位置を同定する。この場合、モバイル端末システムは車両に備え付けられてもよいし、歩行者に所持されていてもよい。モバイル端末システムのさらなる小型軽量化に伴い、このようなサービスはますます現実味を帯びてきている。該サービスを利用することで、危険度の高い人々、例えば高齢者、障害者、病人などを追跡、位置確認することができる。該サービスは、自動車その他の高価な動産や荷物をトラッキングすることにも利用できる。３Ｇ　ＷＡＮネットワーク、ブルートゥースネットワーク、８０２．１１ネットワークその他の無線技術を利用し、ネットワーク媒体や接続ポイントを切り替えてもシームレスな接続性を保つことができるという特性を生かして、上記のサービスを非常に低いコストで実施することが可能である。

【００４５】

このように、本発明は企業ネットワークを延長して、ネットワーク管理者が、信頼性や集中的な管理形態を犠牲にすることなく、モバイル端末のユーザに、アプリケーションへの簡便なアクセスを固定端末のユーザの場合と同様に提供することを可能にする。本ソリューションは、既存の有線ネットワーク規格の長所を制定段階にあるモビリティ規格にも持たせて、既存のネットワーク・アプリケーション上で動作可能なソリューションを生み出すものである。

【００４６】

本発明の他の目的、特徴、および優れた点は、以下に示す記載によって十分分かるであろう。また、本発明の利点は、図面を参照した以下の説明で明白になるであろう。

【００４７】

［好ましい実施の形態の詳細な説明］

図１は、本発明の、モバイル強化ネットワークコンピュータシステム１００を例示している。該ネットワークコンピュータシステム１００は、モビリティ管理サーバ１０２と、一つ以上のモバイル端末システム１０４とを含んでいる。モバイル端末システム１０４は、ローカルエリア・ネットワーク（ＬＡＮ）１０８を通じてモビリティ管理サーバ１０２と通信することができる。モビリティ管理サーバ１０２は、モバイル端末システム１０４のネットワーク・レベルでのプロキシとして機能する。モビリティ管理サーバ１０２はこれを、それぞれのモバイル端末システムの状態を管理することと、ネットワーク・アプリケーションをホスティングしているどのピア・システム１１０とも、モビリティ管理サーバ１０２とモバイル端末システム１０４との間の相互接続が断続的で信頼性の低いものであっても常時接続を維持してゆくために必要な、複雑なセッション管理を行うこととによって成し遂げている。本好ましい実施形態では、モビリティ管理サーバ１０２は、本発明の遠隔プロシージャ・コール・プロトコルおよびインターネット・モビリティ・プロトコルを用いて、モバイル端末システム１０４と通信する。

【００４８】

この場合、モバイル端末システム１０４はモビリティ管理サーバ１０２とアクティブに接

続されるが、該接続は常時アクティブに行われるものではない。例えば、
・複数のモバイル端末システム１０４ａ－１０４ｋが、モビリティ管理サーバ１０２と、モバイル相互接続によって（この場合無線で）通信する。例として、ローカル・エリアもしくはワイドエリア・ネットワーク１０８と無線（あるいは有線）でつながった、従来型の電磁（電波）トランシーバ１０６が考えられる。上記のようなモバイル相互接続により、モバイル端末システム１０４ａ－１０４ｋが、一つのカバーエリア（ｃｏｖｅｒ ａｒｅａ）１０７ａから別のカバーエリア１０７ｋへとローミングすることが可能になる。一般的に、モバイル端末システム１０４が、あるカバーエリア１０７から別のエリアにローミングしたり、最も近いトランシーバ１０６から届く範囲を外れたり、あるいは信号を一時的に遮断（例えばビルの柱の裏側を通るなどで）されたりした場合、一時的に通信が切断される。

【００４９】
・別のモバイル端末システム１０４ｌ、１０４ｍ…が、モビリティ管理サーバ１０２と、ドッキングポートやネットワークケーブルコネクタなどの着脱式（ｎｏｎ－ｐｅｒｍａｎｅｎｔ）有線相互接続１０９を介して通信する。モバイル端末システム１０４は、接続１０９が外れたり、モバイル端末システム１０４の電源が切られたりした場合、一時的にＬＡＮ１０８から切断される。

【００５０】
・さらに別のモバイル端末システム（例えば１０４ｎ）が、モビリティ管理サーバ１０２と、ワイドエリア・ネットワーク、ダイアルアップ・ネットワーク、衛星ネットワーク、インターネット等のネットワーク・トポグラフィ１１１を介してノマディック接続される。一例として、ネットワーク１１１のサービスは断続的なものであってもよく、他の例として、モバイル端末システム１０４がある種類の接続形態から別の種類のものへと移行（例えば、モビリティ管理サーバ１０２と有線相互接続１０９を通じて接続される状態からネットワーク１１１を通じて接続される状態へ移行、もしくはその逆）し、移行時に一時的に接続が切断されるという構成であってもよい。

【００５１】
モバイル端末システム１０４は、標準的なモバイル装置や市販のコンピュータであってよく、例えば、モバイル端末システム１０４は、市販のトランシーバおよび／またはネットワークカードを実装したラップトップコンピュータによって構成される。モバイル端末システム１０４は、標準的なネットワーク・アプリケーションや標準的なＯＳ（オペレーティングシステム）を実行し、標準的なトランスポート・レベル・プロトコル（例えばＴＣＰ／ＩＰ）を利用して、トランスポート層で通信を行うものであってよい。本発明において、さらに、モバイル端末システム１０４がクライアントソフトウェアを実行することで、遠隔プロシージャ・コール・プロトコルおよびインターネット・モビリティ・プロトコルを用いたモビリティ管理サーバ１０２との通信が可能になる。上記両プロトコルは、それらと同様のトランスポート・レベルのプロトコルを利用して伝送される。

【００５２】
モビリティ管理サーバ１０２は、ウィンドウズ（登録商標）ＮＴサーバなどの標準的なサーバによってホスティングされるソフトウェアを含んでいてよい。本好ましい実施形態では、モビリティ管理サーバ１０２は、規格に準拠したクライアント／サーバ型のインテリジェントサーバであって、企業ネットワーク１０８を透過性を有した状態でノマディック環境にまで延長するものである。モビリティ管理サーバ１０２は、数量を限定されないモバイル端末システム１０４それぞれのネットワーク・レベルでのプロキシとして機能するが、モビリティ管理サーバ１０２はこれを、それぞれのモバイル端末システムの状態を管理することと、ネットワーク・アプリケーションをホスティングしているどのピア・システム１１０とも、モバイル端末システム１０４とトランシーバ１０６との間のモバイル相互接続が断続的で信頼性が低いものであっても常時接続を維持してゆくために必要な、複雑なセッション管理を行うこととによって成し遂げている。

【００５３】

例えば、サーバ１０２はどのような標準的（例えばＴＣＰ／ＩＰベースの）ネットワーク・アプリケーションであっても、変更を行うことなしに、モバイル接続を通じて実行させることができる。接続が切断されたり、圏外に出たり、あるいは動作をサスペンドしたりしたモバイル端末システム１０４のセッションは、サーバ１０２によって維持され、該システムが復帰した際にはサーバ１０２が上記セッションをレジュームする。モバイル端末システム１０４が圏外に出てしまったり、シャットダウンしたり、あるいは位置アドレスを変更した場合、モビリティ管理サーバ１０２は、データの受信に関して確認応答し、モバイル端末システムが再度通信可能となるまで要求を待機させることで、モバイル端末システムとピア・システム１１０との接続を維持する。

【００５４】

サーバ１０２はまた、有線ネットワークにおける管理能力をモバイル接続にまで延長する。ネットワークソフトウェア層はそれぞれが互いに独立して動作するので、ソリューションをそれが展開されるそれぞれの環境に合わせてカスタマイズすることが可能である。

【００５５】

一例として、モビリティ管理サーバ１０２が、ローカルエリア・ネットワークやワイドエリア・ネットワークのような標準的な企業ネットワーク１０８と接続され、該ネットワーク１０８は、様々な固定端末システム（例えば一つ以上のホストコンピュータ１１０）１１０と接続される状況が考えられる。モビリティ管理サーバ１０２によって、モバイル端末システム１０４と固定端末システム１１０との間の、連続セッション型データストリームを利用した通信が可能となるが、この通信は、モバイル端末システム１０４が、接続しているネットワーク相互接続とのコンタクトを失ったり、あるネットワーク相互接続１０６、１０９、１１１から別のネットワーク相互接続へと移動したりしても（例えば、無線相互接続の場合、ある無線トランシーバ１０６のカバーエリア１０７から別のカバーエリアにローミングしても）、利用可能となっている。

【００５６】

モバイル端末システム１０４は、モビリティ管理サーバ１０２との結合を、スタートアップ時か、あるいはモバイル端末システムがネットワークサービスを要求した時に確立する。結合が確立されると、モバイル端末システム１０４は、一つ以上のネットワーク・アプリケーションのセッションを、連続的あるいは共時的に始めることができる。モバイル端末システム１０４－モビリティ管理サーバ１０２間の結合確立によって、モバイル端末システムが切断されたり、圏外に出たり、あるいはサスペンドしたりしても、モバイル端末システムにおけるアプリケーションのセッションを維持し、モバイル端末システムの復帰時には該セッションをレジュームすることが可能になる。本好ましい実施形態では上記の処理は完全に自動で行われ、ユーザによる介入は全く必要とされない。

【００５７】

本発明の一例において、モバイル端末システム１０４は、ＵＤＰ／ＩＰのような標準的なトランスポート・プロトコルを用いてモビリティ管理サーバ１０２と通信を行う。標準的なトランスポート・プロトコルを使用することで、モバイル端末システム１０４が、標準的なルータ１１２など企業ネットワーク１０８に既存のインフラを用いて、モビリティ管理サーバ１０２と通信することが可能になる。本発明では、高レベルの遠隔プロシージャ・コール・プロトコルが、トランザクションを、モバイル強化ネットワーク１０８を介して、標準的なトランスポート・プロトコルを使用して送られるメッセージへと生成する。本好ましい実施形態において、上記のようなモバイルＲＰＣメッセージは、モバイル端末システム１０４にて実行されるアプリケーションによって開始された、全てのネットワーク・トランザクションを含んでいるため、モビリティ管理サーバ１０２によって全てを完了させることができる。このことにより、モビリティ管理サーバ１０２とモバイル端末システム１０４とは、ネットワーク媒体の接続性が阻害されているときでも、接続状態情報を常に互いにシンクロさせておくことができる。

【００５８】

モバイル端末システム１０４のそれぞれは、全てのネットワーク活動を傍受し、該ネット

ワーク活動をモバイルＲＰＣプロトコルを通じてモビリティ管理サーバ１０２にリレーするための情報をモバイル端末システム自体に提供するモビリティ管理ソフトウェアクライアントを実行する。本好ましい実施形態では、該モビリティ管理クライアントは、モバイル端末システム１０４に実装されているＯＳ（ウィンドウズ（登録商標）ＮＴ、ウィンドウズ（登録商標）９８、ウィンドウズ（登録商標）９５、ウィンドウズ（登録商標）ＣＥなど）と透過性を有する状態で協働して、クライアント側でのアプリケーションのセッションを、ネットワークとのコンタクトが失われても維持し続ける。

【００５９】

モビリティ管理サーバ１０２は、それぞれのモバイル端末システム１０４の状態を管理し、例えば接続の反対側の端に接続されたホストコンピュータ１１０のような通信相手のピア１０８との連続的な接続を維持するために必要とされる複合的なセッション管理を行う。モバイル端末システム１０４との通信ができなくなったり、モバイル端末システム１０４がサスペンドしたり、あるいはネットワークアドレスを変更したり（例えばあるネットワーク相互接続から別のものにローミングしたり）した場合、モビリティ管理サーバ１０２は、モバイル端末システム１０４とホストシステム１１０などの接続端との間の接続を、データ受信に関して確認応答したり、要求を待機させたりすることによって維持する。このプロキシ機能によって、ピアのアプリケーションは、モバイル端末システム１０４との物理的な接続が絶たれたことを検知することがなくなる。よって、もしモバイル端末システムが一時的に接続を失ったり、あるカバーエリア１０７Ｋ内において、あるネットワーク相互接続１０６Ａから別のネットワーク相互接続１０６Ｋへとローミングしたりした場合でも、（物理的接続が再確立された際に単純に作業をレジュームすることで）モバイル端末システム１０４のアプリケーションと、その結合しているセッションを実行する端との間の連続的な接続を効果的に維持することができる。

【００６０】

モビリティ管理サーバ１０２はまた、モバイル端末システム１０４がセグメント化されたネットワークの様々な部分をローミングする際、異なったネットワークアドレスを受信するという問題を解決するべく、アドレスを管理することが可能となっている。モバイル端末システム１０４はそれぞれ、プライマリネットワーク上での仮想アドレスを有している。該仮想アドレスは、標準的なプロトコルによって、あるいは静的割り当てによって決定されている。モバイル端末システム１０４のそれぞれについて、モビリティ管理サーバ１０２は該システムの現状における実際の（位置）アドレスに対応して仮想アドレスを割り当てる。モバイル端末システム１０４の、あるネットワーク・セグメントから他のセグメントへの移動に伴って、該システム１０４の現在位置アドレスが変更されても、仮想アドレスは、どの接続がアクティブになっていようが、接続時間が長くなっていようが、上記アドレスが静的に割り当てられている限りは一定となる。

【００６１】

よって、モバイル端末システム１０４の位置アドレスの変更は、モビリティ管理サーバ１０２を介して、結合しているホストシステム１１０（および他のピア）におけるセッションを実行する端からは完全に透過性を有する状態となっている。ピア１１０から見えるのは、サーバ１０２によってプロキシされた（不変の）仮想アドレスのみということになる。

【００６２】

本好ましい実施形態では、モビリティ管理サーバ１０２はさらに、コンソール（制御）アプリケーションと包括的なメトリクス（ｅｘｈａｕｓｔｉｖｅ　ｍｅｔｒｉｃｓ）とによる、集中的なシステム管理が可能である。システム管理者は、上記のツールを利用して、遠隔接続を設定、管理し、遠隔接続およびシステムにおける問題を解決することができる。

【００６３】

モビリティ管理サーバ１０２によるプロキシ・サーバ機能によって、ネットワーク・アプリケーション、ユーザ、そして装置にそれぞれ異なった優先レベルを設定することができ

るようになる。これは、モビリティ管理サーバ１０２が有する処理用のリソースが有限であることを鑑みれば、有用なものである。システム管理者が上記のようにモビリティ管理サーバ１０２の設定を変更できることで、システムおよびネットワークのパフォーマンスが全体として向上する。一例として、システム管理者がモビリティ管理サーバ１０２の設定を変更できることにより、音声や映像のストリーミングのようなリアルタイムのアプリケーションに対して、モビリティ管理サーバ１０２のリソースを、あまりリソースを使用しないメールソフトのようなアプリケーションよりも多く割り当てることができる。

【００６４】

詳しく説明すると、モビリティ管理サーバ１０２は、アプリケーション、もしくはＳＮＭＰのような標準的なネットワーク管理プロトコル、ウェブベースの設定インターフェイス、ローカルなユーザインターフェイスなどのアプリケーション・インターフェイスを介して設定される。結合（ａｓｓｏｃｉａｔｉｏｎ）そのものの優先度、および／または、ある結合における複数のアプリケーションの優先度を設定することも可能である。例えば、モビリティ管理サーバ１０２を介して動作する他の結合と関連している結合それぞれの優先度は、ユーザあるいは装置ごとに設定可能である（本実施形態では、ユーザおよびユーザがログインした装置の両方を優先するように設定された場合に、ユーザに関する設定のほうがより優先されるように設定されてもよい）。あるいは結合それぞれについて、アプリケーションの優先度に関し、ネットワーク・アプリケーションごとにいくつかのレベルが設定されていてもよい。本システムでは、優先レベルはいくつでも設定することが可能である。一例として、低、中、高の３つの優先レベルが設定される例が考えられる。

【００６５】

（サーバ／クライアント型ソフトウェア・アーキテクチャの例）

図２に、モバイル端末システム１０４とモビリティ管理サーバ１０２とのソフトウェア・アーキテクチャの一例を図示する。本発明の一例において、モバイル端末システム１０４とモビリティ管理サーバ１０２は標準的なＯＳおよびアプリケーション・ソフトウェアを実行するが、このときに、ほんの少数のコンポーネントを新しく追加するだけで、断続的に接続されるモバイルネットワーク１０８を介した、信頼性が高くかつ効果的、持続的なセッションが実行可能となっている。図２に示すように、モバイル端末システム１０４は、ネットワーク・インターフェイス・ドライバ２００、ＴＣＰ／ＵＤＰトランスポートサポート２０２、トランスポート・ドライバ・インターフェイス（ＴＤＩ）２０４、および一つ以上の従来型のネットワーク・アプリケーション２０８に対するインターフェイスとして使われるソケットＡＰＩ２０６を含む、従来型のＯＳソフトウェアを実行する。従来型のネットワーク・ファイル／プリント・サービス２１０が、従来型のＴＤＩ２０４との通信用にさらに設けられていてもよい。サーバ１０２は、上記と類似した、従来型のネットワーク・インターフェイス・ドライバ２００’、ＴＣＰ／ＵＤＰトランスポートサポート２０２’、トランスポート・ドライバ・インターフェイス（ＴＤＩ）２０４’、および一つ以上の従来型のネットワーク・アプリケーション２０８’に対するインターフェイスとして使われるソケットＡＰＩ２０６’を有していてもよい。モバイル端末システム１０４とモビリティ管理サーバ１０２はそれぞれ、さらにネットワーク／セキュリティ・プロバイダ２３６（モバイル端末システム）、ユーザ／セキュリティ・データベース２３８（サーバ）を備えていてもよい。

【００６６】

本発明では、新規のモバイル・インターセプタ・コンポーネント２１２が、モバイル端末システム１０４のソフトウェア・アーキテクチャにおける、ＴＣＰ／ＵＤＰトランスポートモジュール２０２とトランスポート・ドライバ・インターフェイス（ＴＤＩ）２０４との間に挿入されている。該モバイル・インターセプタ・コンポーネント２１２は、トランスポート・ドライバ・インターフェイス（ＴＤＩ）２０４における特定のコールを傍受して、該コールを、ネットワーク１０８を介し、ＲＰＣおよびインターネット・モビリティ・プロトコル、標準的なＴＣＰ／ＩＰトランスポート・プロトコルを通じてモビリティ管理サーバ１０２へと転送する。こうして、モバイル・インターセプタ２１２は、全てのネ

ットワーク活動を傍受して、サーバ１０２へと転送することができる。該インターセプタ２１２はＯＳと透過性を有する状態で協働するので、モバイル端末システム１０４がネットワーク１０８とのコンタクトを失っても、クライアント側のアプリケーションのセッションはアクティブであり続けることができる。

【００６７】

モバイル・インターセプタ２１２は、トランスポート・ドライバ・インターフェイス２０４とは違うレベルで（例えばソケットＡＰＩ２０６のレベルで）動作することもできるが、ＴＤＩのレベルでモバイル・インターセプタ２１２が動作する、より詳細にはいずれかのトランスポート・プロトコル・インターフェイスにおいて動作することにより、下記のような利点が生まれる。なお、便宜的に、トランスポート・ドライバ・インターフェイスを示す全てのものをＴＤＩという略語で表わすこととする。多くの標準的なＯＳ（例えば、マイクロソフト社のウィンドウズ（登録商標）９５、ウィンドウズ（登録商標）９８、ウィンドウズ（登録商標）ＮＴ、ウィンドウズ（登録商標）ＣＥなど）はＴＤＩインターフェイス２０４を実装しているので、ＯＳのコンポーネントを変更することなく互換性が保証される。さらに、トランスポート・ドライバ・インターフェイス２０４は通常カーネル・レベルのインターフェイスであることから、ユーザモードへの切り替えが必要でなく、これにより性能を向上させることができる。

【００６８】

さらに、ＴＤＩインターフェイス２０４のレベルで作動するモバイル・インターセプタ２１２は、様々な別のネットワーク・アプリケーション２０８（例えば複数の同時に動作しているアプリケーション）に加えて、ネットワークにおけるファイル、プリント、および他のカーネル・モードなどのサービス２１０（インターセプタが例えばソケットＡＰＩ２０６のレベルで動作している場合はそれぞれ別に扱う必要がある）を傍受することができる。

【００６９】

図２Ａに、どのようにインターセプタ２１２が動作するかを示す高レベルのフローチャートの一例を示す。モバイル端末システム１０４のＴＤＩインターフェイス２０４へのコール（ブロック２５０）は、モバイル・インターセプタ２１２によって傍受される（ブロック２５２）。モバイル・インターセプタ２１２によって傍受されたＲＰＣコールはインターネット・モビリティ・プロトコルに準拠してフラグメントにパッケージ化され、該フラグメントはデータグラムとして、ＵＤＰやＴＣＰといった標準的なトランスポート・プロトコルにより、ＬＡＮ、ＷＡＮ等のトランスポート１０８を介してモビリティ管理サーバ１０２へと送られる（ブロック２５２）。モビリティ管理サーバ１０２は受信したＲＰＣデータグラムをアンパックして（ブロック２５４）、要求されたサービスを実行する（例えば、データや応答を、固定端末システム１１０で動作するアプリケーションサーバ処理へと転送することで、モバイル端末システムのアプリケーション２０８のプロキシとして振舞う）。

【００７０】

再び図２に戻って、モビリティ管理サーバ１０２は、従来型のネットワーク・インターフェイス・ドライバ２２２を介して送られる、モバイル端末システム１０４からの、あるいはモバイル端末システム１０４へ向かうメッセージを傍受する、アドレス変換部２２０を有している。例えば、アドレス変換部２２０は、セッションの相手のピア（固定端末システム１１０）からのモバイル端末システム１０４の仮想アドレス宛のメッセージを認識する。モバイル端末システムへの該メッセージはプロキシ・サーバ２２４へと送られる。プロキシ・サーバ２２４は仮想アドレスとメッセージとを待機していたトランザクションに割り当て、該応答を、上記モバイル端末システム１０４の現在位置アドレスへと転送する。

【００７１】

さらに、図２によると、モビリティ管理サーバ１０２は、アドレス変換部（中間ドライバ）２２０とプロキシ・サーバ２２４に加えて、設定マネージャ２２８、操作／ユーザイン

ターフェイス２３０、およびモニタ２３２を有している。設定マネージャ２２８は設定情報とパラメータとを提供して、プロキシ・サーバ２２４が接続の管理を行えるようにする。操作／ユーザインターフェイス２３０とモニタ２３２により、ユーザとプロキシ・サーバ２２４との間のやりとりが可能になる。

【００７２】

（モバイル・インターセプタ）

図３は、本発明のＲＰＣプロトコルとインターネット・モビリティ・プロトコルとをサポートしたモバイル・インターセプタ２１２のソフトウェア・アーキテクチャの一例を示す。本例では、モバイル・インターセプタ２１２は、遠隔プロシージャ・コール・プロトコル・エンジン２４０と、インターネット・モビリティ・プロトコル・エンジン２４４の、二つの機能コンポーネントを有している。モビリティ管理サーバ１０２において動作するプロキシ・サーバ２２４によって、対応するエンジン２４０’、２４４’が用意される。

【００７３】

このように、本好ましい実施形態におけるモバイル・インターセプタ２１２は、モビリティ管理サーバ１０２をそれぞれのモバイル端末システム１０４と接続するための、遠隔プロシージャ・コール・プロトコルと、インターネット・モビリティ・プロトコルとをサポートしている。遠隔プロシージャ・コールは、あるローカルシステムから、離れた別のシステムにおけるプロシージャを発動する処理を可能とするものである。一般的に、ローカルシステムは遠隔のシステムにおいてプロシージャ・コールが実行されたことを感知しない。ＲＰＣプロトコルの利用によって、モバイル端末システム１０４が、アクティブなネットワークセッションを失うことなく、圏外に出たり、操作をサスペンドしたりすることが可能になる。セッションの維持に特別なアプリケーションは必要とされないので、モバイル環境にあるネットワーク１０８において、市販のアプリケーションが何の変更も必要とせずに動作することになる。

【００７４】

ネットワーク・アプリケーションは、一般的に、ウィンドウズソケット（Ｗｉｎｄｏｗｓ　ｓｏｃｋｅｔｓ）のようなアプリケーションレベルのインターフェイスを利用している。アプリケーションレベルのＡＰＩへの単一のコールによって、トランスポート層もしくはメディア・アクセス層における複数の送信および受信データパケットが生成される。優先されるモバイルネットワークで、もし上記のパケットのうちの一つが失われた場合、接続全体の状態が不安定になってセッションは中止されるが、本発明の好ましい実施形態はＲＰＣを備えており、モビリティ管理サーバ１０２とモバイル端末システム１０４とは、物理的な接続が絶たれたときであっても、常に統一された論理リンクを維持するために、接続状態に関する情報を共有している。

【００７５】

本発明のインターネット・モビリティ・プロトコルは、有線のネットワークと無線など他の信頼性の劣るネットワークとの相違点を補償する。フレームサイズとプロトコルのタイミングとが修正されることで、モバイル環境を考慮しないトランスポートと比較して性能が著しく向上し、ネットワークのトラフィックを大幅に減少することができる。このことは、帯域幅に制限がある場合やバッテリーの寿命が問題となる場合に重要となる。

【００７６】

また、本発明のインターネット・モビリティ・プロトコルは、公共の有線ネットワークを通じて、もしくは無線によってモビリティ管理サーバ１０２とモバイル端末システム１０４との間の通信がなされる際の、機密情報のセキュリティを保証する。インターネット・モビリティ・プロトコルは、認証された装置のみが企業ネットワークにアクセスすることを許可することで、基本的なファイアウォールとしての機能を果たす。本発明のインターネット・モビリティ・プロトコルはさらに、モビリティ管理サーバ１０２とモバイル端末システム１０４との間での全ての通信を認証、暗号化することを可能にする。

【００７７】

図３のモバイル端末システム１０４における遠隔プロシージャ・コール・プロトコル・エ

ンジン２４０は、ＴＤＩコールパラメータをマーシャリングし、データをフォーマットし、要求を、ＴＤＩ遠隔プロシージャ・コール・プロトコル・エンジン２４０’がコールを実行するモビリティ管理サーバ１０２まで転送するべく、インターネット・モビリティ・プロトコル・エンジン２４４に送る。モバイル端末システム１０４は、遠隔プロシージャ・コール・プロトコルに基づいてＴＤＩコールパラメータをマーシャリングする。モビリティ管理サーバ１０２のＴＤＩ遠隔プロシージャ・コール・プロトコル・エンジン２４０’が上記のＲＰＣを受信すると、モビリティ管理サーバ１０２はモバイル端末システム１０４に代わってコールを実行する。モビリティ管理サーバ１０２のＴＤＩ遠隔プロシージャ・コール・プロトコル・エンジン２４０’は、接続されたモバイル端末システムそれぞれにおける完全なネットワークの状態を、ピアであるモバイル端末システム１０４のＲＰＣエンジン２４０と共有する。モバイル端末システム１０４の代理としての遠隔プロシージャ・コール実行に加え、サーバのＲＰＣエンジン２４０’は、システムフローの管理、遠隔プロシージャ・コールの解析、仮想アドレスの（アドレス変換機構２２０が行うサービスに対応した）多重化、遠隔プロシージャ・コールのトランザクションの優先順位付け、スケジューリング、ポリシー実行、および融合処理（ｃｏａｌｅｓｃｉｎｇ）を行う。

【００７８】

インターネット・モビリティ・プロトコル・エンジン２４４は、信頼性の高いデータグラムサービス、順位付け、フラグメント化、およびメッセージの再組立を行う。また、設定すれば、認証、データ暗号化、プライバシー保護強化、セキュリティ、そしてスループットのための圧縮も行うことができる。インターネット・モビリティ・プロトコル・エンジン２４４は、電力消費を考慮に入れる必要のある環境において、複数の異なるトランスポートを利用して機能するようになっているので、消費電力管理を意識したものであるとともに、各トランスポートに対して独立となっている。

【００７９】

図３Ａは、モバイル・インターセプタ２１２がモビリティ管理サーバ１０２と通信してＴＤＩコールのやり取りを行う処理を例示する。一般的に、モバイル・インターセプタ２１２のＲＰＣプロトコル・エンジン２４０は、マーシャリングされたＴＤＩコールを、モビリティ管理サーバ１０２へと送るべく、インターネット・モビリティ・プロトコル・エンジン２４４へと転送する。ＲＰＣプロトコル・エンジン２４０は、この作業を、インターネット・モビリティ・プロトコル・エンジン２４４によって管理されるキューにＲＰＣコールを加えることで達成する（ブロック３０２）。帯域幅管理を円滑に行うために、インターネット・モビリティ・プロトコル・エンジン２４４は受信したＲＰＣコールの転送を所定の期間（ＲＰＣ融合タイムアウト期間）遅延させる（ブロック３０４）。一般的に、ＲＰＣ融合タイムアウトは５ミリ秒から１５ミリ秒の間に設定されるが、この数値はユーザによって変更可能である。この遅延によって、ＲＰＣエンジン２４０はＴＤＩコールをインターネット・モビリティ・プロトコル・エンジン２４４のキューに加えて、一つ以上のＲＰＣコールが同一のデータグラム（フラグメント）によって転送されるようにすることができる。

【００８０】

融合タイムアウトが終了するか、ＲＰＣエンジン２４０がそれ以上のＲＰＣコールの受信を拒否する（判定ブロック３０６）ことを決定すると、ＲＰＣエンジン２４０は、インターネット・モビリティ・プロトコル・エンジン２４４に、キューをフラッシュ（ｆｌｕｓｈ）し、ＲＰＣコールを単一のフレームに融合し、該フレームをピアに転送するよう要求する（ブロック３０８）。この融合により伝送回数が減少し、プロトコルのパフォーマンスが高められる。しかし、インターネット・モビリティ・プロトコルは、パフォーマンス最適化のために他の外部基準に基づいてキュー２４４をフラッシュすることもしなければならず、本好ましい実施形態において、単一のＲＰＣ要求でフレーム全体が占められてしまう場合、ＩＭＰ層は自動的に要求をピアに送るよう試みる。

【００８１】

上記のように、モビリティ管理サーバ１０２のプロキシ・サーバはＲＰＣプロトコル・エ

ンジン２４０’とインターネット・モビリティ・プロトコル・エンジン２４４’とを有している。図３Ｂは、モバイル端末システム１０４からインターネット・モビリティ・プロトコル・メッセージ・フレームを受信した際に、モビリティ管理サーバ１０２で実行される処理を例示している。該フレームをモビリティ管理サーバ１０２が受信すると、インターネット・モビリティ・プロトコル・エンジン２４４’は、フレームが（基礎となるトランスポートの最大伝送量の制約で）フラグメント化されていれば再構築し、メッセージの内容を逆多重化して、どのモバイル端末システム１０４からの受信かを決定する。この逆多重化により、インターネット・モビリティ・プロトコル・エンジン２４４’は、遠隔プロシージャ・コール・プロトコル・エンジン２４０’に、的確な結合関連文脈情報（ａｓｓｏｃｉａｔｉｏｎ－ｓｐｅｃｉｆｉｃ ｃｏｎｔｅｘｔ ｉｎｆｏｒｍａｔｉｏｎ）を伝えることができる。

【００８２】

続いて、インターネット・モビリティ・プロトコル・エンジン２４４’は、受信したメッセージを、ＲＰＣ受信示唆システム作業要求（ＲＰＣ ｒｅｃｅｉｖｅ ｉｎｄｉｃａｔｉｏｎ ｓｙｓｔｅｍ ｗｏｒｋ ｒｅｑｕｅｓｔ）３５４にして、該作業要求と結合関連文脈情報を、モビリティ管理サーバ１０２のＲＰＣプロトコル・エンジン２４０’に与える。ＲＰＣプロトコル・エンジン２４０’が作業要求３５２を受信すると、該エンジン２４０’は作業要求３５２を結合関連作業キュー３５６に加え、次に予定済みの要求をグローバルキュー３５８に送ることによって、結合処理のスケジューリングが行われる。続いて、ＲＰＣエンジン２４０’のメイン作業スレッドに、作業が実行可能となったことが伝えられる。該メインスレッドが作業を始めると、グローバルキュー３５８がポーリングされて、待機状態となっているスケジューリングされた結合処理が確認される。その後、メインスレッドは該イベントを待機状態から外し、結合関連作業キュー３５６が処理される。

【００８３】

結合関連作業キュー３５６から、上記メインスレッドはそれまで待機していたＲＰＣ受信示唆作業要求を見つけ出す。次に、メインスレッドはＲＰＣ受信示唆作業要求３５６をキューから外し、該要求を解析する。図３Ａを参照して説明した上記融合処理により、モビリティ管理サーバ１０２は、それぞれのデータグラムに内包された複数のＲＰＣトランザクションを頻繁に受信することになる。モビリティ管理サーバ１０２は次に、該ＲＰＣトランザクションをそれぞれ逆多重化して別個の遠隔プロシージャ・コールに戻し、そして要求された機能をモバイル端末システム１０４に代わって実行する。パフォーマンス向上のため、ＲＰＣエンジン２４０’に、ＲＰＣメッセージ解析処理中の先読みメカニズムを備えさせて、ＲＰＣエンジン２４０’が、要求されたトランザクションのうちのいくつかを同時に実行できるか否か（パイプライン処理できるか否か）を確認してもよい。

【００８４】

（ＲＰＣエンジン２４０’によるＲＰＣ結合の実行手法）

図４は、結合作業キュー３５６に加えられたＲＰＣ結合を実行する処理を例示したフローチャートである。ＲＰＣ結合の実行が予定されている時、ＲＰＣプロトコル・エンジン２４０’（状態機械として設けられていてもよい）のメインスレッドは、グローバルネットワークキュー３５８からの作業要求を待機状態から外し、作業要求の種類を決定する。

【００８５】

本実施形態のＲＰＣ作業要求は、以下のような６つの基本的な種類に分けられる。

【００８６】

・スケジューリング要求（ｓｃｈｅｄｕｌｅ ｒｅｑｕｅｓｔ）
・接続示唆
・切断示唆
・ローカル結合中止（ｌｏｃａｌ ｔｅｒｍｉｎａｔｅ ａｓｓｏｃｉａｔｉｏｎ）
・「リソース使用可」要求（”ｒｅｓｏｕｒｃｅ ａｖａｉｌａｂｌｅ” ｒｅｑｕｅｓｔ）

・ｐｉｎｇ休止タイムアウト（ｐｉｎｇ　ｉｎａｃｔｉｖｉｔｙ　ｔｉｍｅｏｕｔ）
ＲＰＣプロトコル・エンジン２４０’は、上記の様々な種類の要求をそれぞれ別の方法で取り扱う。ＲＰＣプロトコル・エンジン２４０’は要求の種類（グローバルキュー３５８に記憶された、要求に関する情報によって識別される）をテストして、該要求をどう処理するかを決定する。

【００８７】
作業要求の種類が「スケジューリング要求」である場合（判定ブロック３６０）、ＲＰＣプロトコル・エンジン２４０’はどの結合が予定されているかを判別する（ブロック３６２）。ＲＰＣプロトコル・エンジン２４０’はこの判別を、グローバルキュー３５８に記憶されている情報に基づいて実行することができる。上記判別がなされると、ＲＰＣプロトコル・エンジン２４０’が、それぞれが対応する要求を記憶している結合作業キュー３５６（１）～３５６（ｎ）のうちの一つを特定することが可能となる。ＲＰＣエンジン２４０’によって、対応する結合制御ブロックが取得され（ブロック３６２）、結合作業処理タスク（Ｐｒｏｃｅｓｓ　Ａｓｓｏｃｉａｔｉｏｎ　Ｗｏｒｋ　ｔａｓｋ）３６４が呼び出されて、上記の特定された作業キュー３５６における作業の処理が始められる。

【００８８】
図５は、図４の「結合作業処理」タスク３６４によって実行されるステップを例示している。結合が特定されると、「結合作業処理」タスク３６４が呼び出され、対応する結合作業キュー３５６内の作業が処理される。待機状態から外された作業要求（ブロック３９０）がＲＰＣ受信要求（判定ブロック３９２）である場合、該ＲＰＣ受信要求はＲＰＣ構文解析部（ｐａｒｓｅｒ）に送られ、処理される（ブロック３９４）。あるいは、待機状態から外された作業要求（ブロック３９０）が、保留中の（ｐｅｎｄｉｎｇ）受信要求である場合（判定ブロック３９６）、ＲＰＣエンジン２４０’はＴＤＩ２０４’に、アプリケーションによる接続の代わりにデータを受信し始めるよう要求する（ブロック３９８）。上記待機状態から外された作業要求が、保留中の接続要求である場合（判定ブロック４００）、ＲＰＣエンジン２４０’はＴＤＩ２０４’に、アプリケーション用途ＴＣＰ（あるいは他のトランスポート・プロトコル）接続要求を発するよう要求し（ブロック４０２）、ＴＤＩ層２０４’からの応答を待つ。ＴＤＩ２０４’による要求が完了すると、該要求の状態が決定されて、元の要求エンティティへと報告が戻される。パフォーマンス向上のため、ＲＰＣエンジン２４０’は、実際に要求を発しているピアへエラーを通知する前に、要求を結合関連作業キュー（３５６）へと戻すことで、接続要求処理を複数回行うようになっていてもよい。この処理も、ネットワーク帯域幅およびリソースの消費を減らすためのものである。

【００８９】
上記の処理は、「スケジューリング重み付け」（ｓｃｈｅｄｕｌｉｎｇ　ｗｅｉｇｈｔ　ｃｏｍｐｌｅｔｅ）テスト（ブロック４０４）にパスするまで繰り返される。本例では、スケジューリング重みは、どれだけの作業要求が待機状態から外されて、上記の特定の結合がどれだけ処理されるかを決定するのに使われる。該スケジューリング重みは設定マネージャ２２８によって設定されるパラメータであり、結合接続示唆が検出されたときに取得される（図４、ブロック３７２）。この数値はユーザごとに、あるいは物理的な意味での装置ごとに設定可能である。

【００９０】
ＲＰＣエンジンが結合作業キュー３５６の作業を（少なくとも一時的に）終了した後、ディスパッチ・キューの処理に移ってもよい（ブロック４０６）（詳細は以下）。結合の作業キュー３５６における作業の処理後、ＲＰＣエンジン２４０’は、グローバル作業キュー３５８に新たなスケジューリング要求を発信して、後で実行される結合のスケジューリングを再び行う（図４の判定ブロック３６６、ブロック３６８、図５の判定ブロック４０８、ブロック４１０）。

【００９１】
再度図４を参照すれば、ＲＰＣ作業要求が「接続示唆」の場合（判定ブロック３７０）、

ＲＰＣエンジン２４０’に、モバイル・ピア（通常はモバイル端末システム１０４だがこれに限らない）との新たな接続のインスタンスを作成せよという要求が入る。一例として、上記接続示唆により、接続を開始しようとしているピアの装置に関する以下のような情報がＲＰＣエンジン２４０’に与えられる。

【００９２】

・装置の物理的識別子
・該装置にログインしているユーザ名
・ピアの装置のアドレス
・ピアのＲＰＣエンジン２４０からの、付加的な接続データ

接続示唆（判定ブロック３７０）を受け、ＲＰＣエンジン２４０は上記のパラメータをもって設定マネージャ２２８をコールする。設定マネージャ２２８は該パラメータを用いて、上記新規接続の設定を確定する。この設定（例えば結合スケジューリング重みや、上記に加えてデフォルトでないスケジューリング特性を要するアプリケーション全てのリストなど）は、ＲＰＣエンジン２４０’に戻されて記憶、実行される。そしてＲＰＣエンジン２４０’は新規の結合を開始し、新規の結合制御ブロックを形成する（ブロック３７２）。図５Ａが示すように、以下のような動作が実行されてもよい。

【００９３】

・結合制御ブロックを割り当てる（ブロック３７２Ａ）
・システム全体のリソースをデフォルトにまで初期化する（ブロック３７２Ｂ）
・現在の設定をオーバーライドする（ブロック３７２Ｃ）
・フラグを初期化する（ブロック３７２Ｄ）
・結合特有作業キューを初期化する（ブロック３７２Ｅ）
・結合のオブジェクト・ハッシュ・テーブルを初期化する（ブロック３７２Ｆ）
・融合タイマーを初期化する（ブロック３７２Ｇ）
・結合制御ブロックをセッションテーブルに挿入する（ブロック３７２Ｈ）

インターネット・モビリティ・プロトコル・エンジン２４４’が結合を終了しなければならないと判断すると、「切断示唆」が該インターネット・モビリティ・プロトコル・エンジン２４４’からＲＰＣエンジン２４０’に出される。ＲＰＣエンジン２４０’はこの切断示唆をテストし（ブロック３７４）、示唆に応じて結合を停止し、結合制御ブロックを破棄する（ブロック３７６）。図５Ｂに示されるように、以下のステップが実行されてもよい。

【００９４】

・待機中の作業がこれ以上処理されないように、停止される結合をマークする（ブロック３７６Ａ）
・処理を含む、結合されている全ての結合オブジェクトをクローズする（ブロック３７６Ｂ）
・作業キューにある全てのエレメントを開放する（ブロック３７６Ｃ）
・融合タイマーが動作中ならば停止する（ブロック３７６Ｄ）
・結合制御ブロックの参照カウントを減少させる（ブロック３７６Ｅ）
・参照カウントが０の場合、（ブロック３７６Ｆにてテストされる）
・結合関連作業キューを破棄し、
・オブジェクト・ハッシュ・テーブルを破棄し、
・融合タイマーを破棄し、
・結合テーブルから結合制御ブロックを取り除き、
・制御ブロックを開放する（３７６Ｇ）

システム１０２が結合終了の必要性を認識すると、「セッション中止」要求が出される。この要求はシステム管理者、ＯＳ、あるいはアプリケーションから発される。ＲＰＣエンジン２４０’は、セッション中止要求を上記切断示唆と同様に扱う（判定ブロック３７８、ブロック３７６）。

【００９５】

本好ましい実施形態では、ＲＰＣエンジン２４０’とインターネット・モビリティ・プロトコル・エンジン２４４’との間のインターフェイスが、クレジット（ｃｒｅｄｉｔｓ）を基にフロー制御メカニズムを特定する。ある単一のスレッドが別のスレッドに作業要求を通知するたびに、コールされたスレッドは作業キューに残っているクレジットの数に戻る。キューが満杯であればクレジットのカウントは０になるが、慣例として、コールする側のスレッドは、クレジットのカウントが０ならばそれ以上作業通知をしない。よって、ユーザによって設定可能あるいは所定の最低点（ｌｏｗ－ｗａｔｅｒ　ｍａｒｋ）をキューに付けて、待機中であった作業が処理されてリソースに余裕ができたことを、上記コールする側のスレッドに伝えるような構成が必要となる。これが、「リソース使用可」作業示唆が設けられている理由である（判定ブロック３８０にてテストされる）。図５Ｃが示すように、クレジットのカウントが０になったとき、以下のようなステップが実行されてもよい。

【００９６】

・ＲＰＣ＿ＬＭＰＱ＿ＳＥＮＤ＿ＦＬＡＧと設定して、結合に「ロー・マーク保留」（ｌｏｗ　ｍａｒｋ　ｐｅｎｄｉｎｇ）とマークを付ける（ブロック３７９Ａ）。この状態になったら、

・受信された全てのデータグラムを破棄する（ブロック３７９Ｂ）

・データ受信を拒否して全ての受信されたストリーム・イベントを抑える（ブロック３７９Ｃ）（これにより、ＴＣＰ他のトランスポート受信ウィンドウが結果的に閉じられ、固定端末システム１１０とモビリティ管理サーバ１０２との間のフロー制御がなされる。復帰の前に、本好ましい実施形態では「保留受信要求」（ｐｅｎｄｉｎｇ　ｒｅｃｅｉｖｅ　ｒｅｑｕｅｓｔ）を結合関連作業キュー３５６の前に押し込むので、保留中のストリーム受信イベントの処理（ｏｕｔｓｔａｎｄｉｎｇ　ｓｔｒｅａｍ　ｒｅｃｅｉｖｅ　ｅｖｅｎｔ　ｐｒｏｃｅｓｓｉｎｇ）が、リソースが利用可能になると直ちに継続される）。

【００９７】

・全ての受信された接続イベントが、受動接続のために拒否される（ブロック３７９Ｄ）「リソース使用可」示唆をＲＰＣエンジン２４０’が受け取ると（図４、判定ブロック３８０）、該ＲＰＣエンジンは、結合された結合作業キュー３５６に待機中の作業があるか否かを判定する。もしあれば、ＲＰＣエンジンは該結合についてグローバル作業キュー３５８に通知して、上記結合作業キュー３５６が動作の優先権を持つことをマークしておく（ブロック３８２）。保留中の受信要求が、結合が保留受信要求状態にある期間に通知された場合、この期間中に処理される（本好ましい実施形態では、ＲＰＣエンジン２４０’は、この処理中も全ての受信された接続要求を受け入れる）。

【００９８】

再度図４を参照すると、ＲＰＣエンジン２４０’が、モビリティ管理サーバ１０２の「ｐｉｎｇ」に使われるチャネルが所定の期間にわたって休止していると判定した場合（判定ブロック３８４）、該チャネルは閉じられ、リソースは開放されてシステムに復帰し、他の処理に使用される（ブロック３８６）。

【００９９】

（ＲＰＣ構文解析と優先キュー）

再度図５を参照すると、ＲＰＣエンジンがＲＰＣ受信要求をその受信に際して構文解析することは前述した（ブロック３９２、３９４参照）。本好ましい実施形態において、構文解析は以下の点で必要とされる。即ち、受信された単一のデータグラムが複数のＲＰＣコールを含む可能性があるからであり、また、ＲＰＣコールがインターネット・モビリティ・プロトコルのデータグラムにおける複数のフラグメントにまで広がっている可能性があるからである。ＲＰＣ受信作業要求５００のフォーマットの例が図６に示される。ＲＰＣ受信作業要求のそれぞれは、少なくともメイン・フラグメント５０２（１）を有し、加えて任意の数の付加的フラグメント５０２（２）～５０２（Ｎ）を有していることもある。メイン・フラグメント５０２（１）は、作業要求構造ヘッダ（ｗｏｒｋ　ｒｅｑｕｅｓｔ　ｓｔｒｕｃｔｕｒｅ　ｈｅａｄｅｒ）５０３と、受信オーバーレイ５０４とを有してい

る。受信オーバーレイ５０４は、インターネット・モビリティ・プロトコル・エンジン２４４によってメイン・フラグメント５０２（１）の先頭に設けられた受信オーバーレイである。この受信オーバーレイ５０４には、ｐユーザデータと呼ばれる、作業要求５００内で最初のＲＰＣコール５０６（１）を指し示す構造メンバがある。

【０１００】

図６の例に、複数のＲＰＣコール５０６（１）、５０６（２）…５０６（８）を含む、作業要求５００が図示されている。図６の例が示すように、ＲＰＣ作業要求５００は、メモリの隣接するブロックや単一のフラグメント５０２に含まれていなくてもよい。同例において、第二フラグメント５０２（２）と第三フラグメント５０２（３）とは、リンクしたリスト中でメイン・フラグメント５０２（１）に連鎖されている。

【０１０１】

よって、上記例のＲＰＣパーサ３９４は以下の境界条件を取り扱う。

【０１０２】

・ＲＰＣ受信要求５００それぞれが一つ以上のＲＰＣコールを含んでもよい
・一つ以上のＲＰＣコール５０６が、単一のフラグメント５０２中に存在してもよい
・ＲＰＣコール５０６それぞれが、フラグメント５０２中に完全に含まれていてもよい
・ＲＰＣコール５０６それぞれは、一つ以上のフラグメント５０２にまたがっていてもよい

図７は、ＲＰＣ受信作業要求５００を解析するＲＰＣ構文解析部３９４を例示している。本例において、ＲＰＣ構文解析部３９４は作業要求中の第一フラグメント５０２（１）を取得し、該フラグメント中の第一ＲＰＣコール５０６（１）を取得し、そしてＲＰＣコールを解析する。構文解析部３９４はＲＰＣ受信作業要求５００中を進んで、ＲＰＣコール５０６それぞれを処理していく。ＲＰＣ受信作業要求５００のフラグメント５０２（１）の残りのフラグメント・バイト数が、ＲＰＣヘッダ５０３のサイズよりも多い場合、構文解析部３９４は、ＲＰＣコールが完全にＲＰＣフラグメント５０２に含まれていて、処理を実行してもよいかどうかを判定する（該判定は、ＲＰＣコールの長さが残りのフラグメント・バイト数より大きいかどうかをテストすることでなされてもよい）。ＲＰＣコールの種類が連鎖例外である場合、ＲＰＣコールはＲＰＣ構文解析部３９４のアップデートを行うことになる。プロキシ・サーバ２２４において、連鎖例外であるＲＰＣコールは「データグラム送信」と「ストリーム送信」だけである。この連鎖例外プロシージャによって、ＲＰＣエンジンは、ＲＰＣ送信コールのためにメモリ記述子リストを共に連鎖することによるフラグメントコピーを避けることができる。

【０１０３】

構文解析部３９４がＲＰＣコールの種類を判別すると、ＲＰＣ情報の始めへのポインタは、実行のためにＲＰＣエンジン２４０へと転送される。ＲＰＣエンジンは、実行のために全てのＴＤＩプロシージャ・コールにそれぞれ別の優先順位をつける。最も優先度の高いコールは、ＲＰＣディスパッチャ３９５へ転送され、直ちに処理される。これより下の優先順位のコールは、後に処理されるために全てディスパッチ・キュー５１０にディスパッチされる。ディスパッチ・キュー５１０はそれぞれ離散的な優先度を表わしている。

【０１０４】

本実施形態では、モバイル・アプリケーションは、「オープンアドレス」オブジェクトおよび「オープン接続」オブジェクト機能を、他のＴＤＩネットワーキング機能を実行する前に呼び出す。よってシステムは、「オープンアドレス」オブジェクトおよび「オープン接続」オブジェクトの呼び出し中に、アプリケーションレベルの優先順位を割り当てることになる。例示の実施形態では、アドレスまたは接続オブジェクトに優先順位が付けられると、該オブジェクトに関連する全てのコールが、その割り当てられた優先順位中に実行される。

【０１０５】

例えば、ＲＰＣコールがＴＤＩオープンアドレスオブジェクト要求あるいはＴＤＩオープン接続オブジェクト要求である場合、該コールは直ちに実行されるべくＲＰＣディスパッ

チャ３９５に送られる。オープンアドレスおよびオープン接続オブジェクトＲＰＣコールにより、上記の接続示唆中に行われる設定要求の間に設定マネージャ２２８からもたらされる情報との対応に用いられる、処理ＩＤあるいは処理名へのアクセスが可能になる。該処理ＩＤあるいは処理名は、アドレスあるいは接続オブジェクトの設定を得るために用いられる。

【０１０６】

本好ましい実施形態では、全てのＲＰＣコールは、パラメータとして少なくともアドレスオブジェクトまたは接続オブジェクトを有している。コールが実行されると、そのオブジェクトに割り当てられた優先度が、ＲＰＣコールの優先度とされる。アドレスあるいは接続オブジェクトに割り当てられた設定により、実行される、対応するＲＰＣコール全ての優先度が決定される。例えば、割り当てられた優先度が「高」の場合、ディスパッチ・キュー５１０にディスパッチされることなく、全てのＲＰＣコールが直ちに実行される。割り当てられた優先度が「１」の場合、全てのＲＰＣコールが、ディスパッチ・キュー５１０（１）に加えられる。

【０１０７】

再度図５を参照すると、「結合作業処理」（ｐｒｏｃｅｓｓ　ａｓｓｏｃｉａｔｉｏｎ　ｗｏｒｋ）タスク３６４の処理が、予定された結合作業量の実行を完了すると（判定ブロック４０４）、ディスパッチ・キューがサービスを要求しているか否かが確認される（ブロック４０６）。図８は、図７に示されるディスパッチ・キュー５１０の処理のための「ディスパッチ・キュー処理」（ブロック４０６、図６）によって実行されるステップを例示したフローチャートである。

【０１０８】

この例では、ディスパッチ・キュー５１０は優先度が最高のキュー（本例では５１０（１））から処理される（ブロック４０８）。ディスパッチ・キュー５１０にはそれぞれ重み係数が設定されている。この重み係数は、モバイル端末システム１０４とモビリティ管理サーバ１０２との結合が確立された際に、設定マネージャ２２８によって返される設定パラメータである。例を挙げれば、優先度の低いディスパッチ・キュー５１０の重み係数は４であり、優先度が中程度のディスパッチ・キュー５１０重み係数は８である。本例では、優先度の高いＲＰＣコールは解析後直ちに処理されるため、重み係数を有していない。

【０１０９】

ＲＰＣエンジン２４０’は、現在のキューより始めて、ＲＰＣコールをキューから外していき、キューが空になるか、ＲＰＣコールのキュー重み番号（ｑｕｅｕｅ　ｗｅｉｇｈｔ　ｎｕｍｂｅｒ）が処理されるまでループする（ブロック４１２－４１６）。キューから外されたＲＰＣコールそれぞれに対して、その実行のためにＲＰＣディスパッチャ３９５が呼び出される。ＲＰＣディスパッチャ３９５はモバイル端末システム１０４の代理としてプロシージャ・コール（ｐｒｏｃｅｄｕｒａｌ　ｃａｌｌ）を実行し、応答を要求している上記ＲＰＣコールに対するモバイル端末システムからの応答を生成する。

【０１１０】

上記ループを出た後、キューに残っている作業がまだある場合（判定ブロック４１８）、該キューが再実行を要することがマークされる（ブロック４２０）。ループを出ることで、上記システムはプロセッサを次に低い優先度のキューに移らせる（ブロック４２４、４１０）。これにより、ある特定のキューにどれだけの作業が割り振られていても、どの優先レベルにも実行される可能性が与えられることになる。システムは次のキューのサービスに移り、全てのキューが処理されるまで処理を繰り返す。全てのキューの処理が終了すると、システムは実行を要するマークがついたキューがあるか否かを判定して、もしあれば、スケジュール要求がグローバル作業キューに送られて、結合の再実行が予定されることになる。結合の再実行は、図４の「グローバル作業処理」ルーチンにおいて予定される。上記のアプローチにより、プロセッサが、処理すべき作業を有する他の結合にも実行の機会を与えることになる。キューそれぞれに重み係数を割り振ることで、システムは、モビリティ管理サーバ１０２のＣＰＵへのアクセスが優先レベルに応じて許可されるように

調整されてもよい。これにより、優先度の高いキューは、先に実行されるだけでなく、さらに容易にＣＰＵにアクセスできるようになる。

【０１１１】

（モビリティ管理サーバのＲＰＣ応答）

上記では、どのように遠隔プロシージャ・コールがモバイル端末システム１０４からモビリティ管理サーバ１０２に送られて実行されるかを説明した。この型のＲＰＣコールに加え、モビリティ管理サーバ１０２のＲＰＣエンジン２４０’はＲＰＣイベントとＲＰＣ受信応答とをサポートしている。結合関連接続ピア（通常は固定端末システム１１０）の活動の結果として、ＲＰＣメッセージが非同期的に生成される。モビリティ管理サーバ１０２のＲＰＣエンジン２４０’は、ＲＰＣディスパッチャ３９５によって実行されるＲＰＣトランザクションを完了する。完了が成功したからといって、全てのＲＰＣコールが応答を要求するわけではないが、応答を要求する一部のＲＰＣコールにより、ＲＰＣディスパッチャ３９５は適切な応答を作成し、インターネット・モビリティ・プロトコル・エンジン２４４’に知らせ、そして該応答はピアのモバイル端末システム１０４に返される。ＲＰＣコールが失敗したときは、全てのＲＰＣコールが応答を生成する（上記においてＲＰＣ受信応答は例外）。

【０１１２】

ＲＰＣイベントは、結合関連接続（通常は固定端末システム１１０）によるネットワーク１０８の活動の結果として発信される。本好ましい実施形態では、こうしたＲＰＣイベントメッセージはモビリティ管理サーバ１０２によってプロキシされ、モバイル端末システム１０４へ送られる。本好ましい実施形態のモビリティ管理サーバ１０２は、以下のＲＰＣイベントコールをサポートする。

【０１１３】

・切断イベント（結合特定接続ピア（通常は固定端末システム１１０）がトランスポート・レベルで切断要求を出した場合に生起する。該要求はモバイル端末システム１０４に代わってプロキシ・サーバ２２４に受信され、プロキシ・サーバ２２４は切断イベントをモバイル端末システムに送る）

・ストリーム受信イベント（結合特定接続ピア（通常は固定端末システム１１０）がストリームデータをモバイル端末システム１０４へ送る際に生起。プロキシ・サーバ２２４はモバイル端末システム１０４に代わって該データを受け取り、受信応答（Ｒｅｃｅｉｖｅ　Ｒｅｓｐｏｎｓｅ）として該データをモバイル端末システムに送る）

・データグラム受信イベント（結合特定ポータル（ａｓｓｏｃｉａｔｉｏｎ－ｓｐｅｃｉｆｉｃ　ｐｏｒｔａｌ）のいずれかが、ネットワーク・ピア（通常は固定端末システム１１０）から送られ、モビリティ管理サーバ１０２を経由してモバイル端末システム１０４へ送られることになっている、データグラムを受け取る際に生起する。プロキシ・サーバ２２４はモバイル端末システム１０４に代わって該データグラムを受け入れ、データグラム受信イベントという形で該データグラムをモバイル端末システム１０４へ転送する）

・接続イベント（結合特定リスニングポータル（ａｓｓｏｃｉａｔｉｏｎ－ｓｐｅｃｉｆｉｃ　ｌｉｓｔｅｎｉｎｇ　ｐｏｒｔａｌ）が、トランスポート層とモバイル端末システム１０４との終端間接続を確立しようとする際、該結合特定リスニングポータルがトランスポート層接続要求（通常は固定端末システム１１０から）を受信するときに生起する。プロキシ・サーバ２２４はモバイル端末システムに代わって接続要求を受け入れ、接続イベントＲＰＣコールを生成してモバイル端末システムに送る）

図９は、ＲＰＣエンジン２４０’が、いかにしてプロキシ・サーバによって生成されたＲＰＣコールを扱うかということを示している。優先度の高いアドレスと接続オブジェクトに対し、ＲＰＣエンジン２４０’は、直ちに送信要求をインターネット・モビリティ・プロトコル・エンジン２４４’に向けてディスパッチする。該送信要求によって、ＲＰＣメッセージがピアのモバイル端末システム１０４に転送される。優先度の低いオブジェクトに対しては、インターネット・モビリティ・プロトコル・エンジン２４４の送信要求が適切な優先キュー５１０’に通知される。結合の実行が予定されていない場合、スケジュー

ル要求もグローバルキュー３５８’に通知される。インターネット・モビリティ・プロトコルの送信要求は、最終的に、図５および８を参照して既に説明したディスパッチ・キューの処理の際に実行される。

【０１１４】

（インターネット・モビリティ・プロトコルの例）

本発明のインターネット・モビリティ・プロトコルは、メッセージ指向で接続ベースのプロトコルであり、配信保証、（再）オーダ検出（（ｒｅ）ｏｒｄｅｒ　ｄｅｔｅｃｔｉｏｎ）、ロス回復が可能である。さらに、他の従来型接続指向プロトコル（即ちＴＣＰ）とは違い、複数の別のデータストリームを単一のチャネルにまとめることが可能になっており、保証されたデータ、信頼性の低いデータ、そして新規のメッセージ指向で信頼性の高いデータを、単一の仮想チャネルによって同時にネットワークをトラバースさせることができる。この新たなメッセージ指向のサービスのレベルによって、インターネット・モビリティ・プロトコルのピアが所定のプログラムデータユニットを確認したときに、リクエスト部に通知が行われることになる。

【０１１５】

本発明のインターネット・モビリティ・プロトコルは、既存のネットワーク・トポロジや技術にオーバーレイできるように設計されている。下層のネットワークアーキテクチャに左右されないので、該インターネット・モビリティ・プロトコルは汎用性を有する。パケット化されたデータが二つのピアの間を行き来できさえすれば、インターネット・モビリティ・プロトコルの使用が可能である。ノードそれぞれにおけるネットワークの現在地点（ＰＯＰ）あるいはネットワーク・インフラは、物理的境界、特定のポリシー、あるいは帯域幅の制限がある場合を除き、データの流れに影響を与えることなく変更可能である。

【０１１６】

インターネット・モビリティ・プロトコルは上層の助けを借りて様々なソースからのデータを融合し、下層のデータグラム機能を利用して該データを行き来させる。独立したデータユニットがそれぞれ上層から来ると、インターネット・モビリティ・プロトコルは該データユニットを単一のストリームとして、その後伝送する。該データユニットは次に既存のネットワークを通じてピアに送られ、受信時に、上層の助力によって上記ストリームが非多重化されて、元の独立した複数のデータユニットに戻る。これにより、伝送時にはその都度最大限のネットワークフレームが生成されて、帯域幅の利用を最適化することができる。さらに、上記により、帯域幅を最大限利用できるようにチャネルを調整することができるという効果を奏するとともに、全てのセッションレベルの接続に適用可能なパラメータを持たせることが可能となる。

【０１１７】

あるチャネルが不十分であるといった稀なケースでも、インターネット・モビリティ・プロトコルはピア間に複数のチャネルを確立することができる。これにより、データの優先順位付けや、（下層のネットワークがサポートしていれば）サービス品質の保証も可能になる。

【０１１８】

さらに、インターネット・モビリティ・プロトコルは、動的に選択可能で保証されたレベルのサービス、あるいは信頼性の低いレベルのサービスに対しても考慮されている。例えば、伝送されるプロトコル・データ・ユニットをそれぞれ、有効期間（ｖａｌｉｄｉｔｙ　ｔｉｍｅ　ｐｅｒｉｏｄ）か再伝送試行回数、もしくはその両方で制限して待機させることができる。インターネット・モビリティ・プロトコルは、どちらかの閾値に達するとデータユニットを有効期限切れとし、その後の伝送の試行から排除する。

【０１１９】

インターネット・モビリティ・プロトコルの、付加プロトコルとしてのオーバーヘッドは、可変長ヘッダの採用により最小限に抑えられている。該ヘッダのサイズはフレームの種類や任意フィールド（ｏｐｔｉｏｎａｌ　ｆｉｅｌｄ）によって決定される。該任意フィールドは、ある特定のオーダにおいて追加されて、受信側による解析を容易にし、その存

在はヘッダ・フラグ・フィールド（ｈｅａｄｅｒ　ｆｌａｇ　ｆｉｅｌｄ）のビットによって示される。ピアによる通信に必要な他の制御および設定情報は、全てバンド内制御チャネルを通過することができる。送信されるべき制御情報は全て、任意のアプリケーションレベルのプロトコル・データ・ユニットの前のフレームに加えられる。受信側では制御情報を処理して、次にペイロードの残りを上層に送る。

【０１２０】

エラー発生の確率が比較的高い、信頼性の低いネットワークにおける稼動が想定されているため、インターネット・モビリティ・プロトコルでは、データの完全性を保証し、ネットワークのパフォーマンスを最高まで引き出すべく、数々の技術が使われている。データの完全性を保証するため、フレッチャー・チェックサム・アルゴリズム（Ｆｌｅｔｃｈｅｒ　ｃｈｅｃｋｓｕｍ　ａｌｇｏｒｉｔｈｍ）が、誤ったフレームの検出用に使われている。このアルゴリズムは効率性と高い検出能力を買われて採用され、ビットエラーのみならずビットの並び替えも検出できる。ただし、上記アルゴリズムの代わりに他のチェックサム・アルゴリズムを採用してもよい。

【０１２１】

シーケンス番号は、オーダされたデータの配信を保証するために使われるが、インターネット・モビリティ・プロトコルにおけるシーケンス番号は、ＴＣＰの場合のようにデータのそれぞれのバイトを表わしているわけではない。一例としては、上記シーケンス番号は、最大６５５３５バイト（インターネット・モビリティ・プロトコル・ヘッダを含む）までの、データのフレームを表現している。該シーケンス番号は３２ビットなど適当な長さのものであって、制限された期間内での高い帯域幅のリンクにおいてラップアラウンドを生じさせないようにしている。

【０１２２】

上記の能力にデータの有効期限切れ機能を組み合わせたとき、再伝送（再試行）されたフレームが含んでいるデータ量が、送信側で生成された前バージョンよりも少ないことがある。フレームＩＤを設定して最新のバージョンのフレームが検知することも可能だが、本好ましい実施形態ではデータが追加されることは絶対に無く、削除されたそれぞれの要素がプロトコル・データ・ユニット全体であるため、上記の手法はシーケンス保証には必ずしも必要でない。一例として、インターネット・モビリティ・プロトコルは受信したある特定のフレームを、該フレームの異なったバージョンのものがいくら伝送されてきても、最初の一回目のみ処理する。新規のユーザ・ペイロードを運搬する、生成されたフレームそれぞれには、固有のシーケンス番号が割り振られる。

【０１２３】

スライディング・ウィンドウ技術の採用によってパフォーマンスが向上し、ピアにデータ受信の確認応答を要求する前に、複数のフレームを保留にできる（伝送できる）。適切なタイミングでのデータ配信を保証するため、肯定応答とタイマーベースの再伝送スキームとが採用されている。さらにチャネルの利用を最適化すべく、選択的な確認応答メカニズムが採用され、ネットワーク接続のロスが多い時間帯もしくは混雑した時間帯における、失われたフレームの迅速な再伝送と、素早い回復とが達成される。一例において、この選択的な確認応答メカニズムは、ヘッダに含まれる付加的なビットフィールドとして表わされる。

【０１２４】

輻輳回避アルゴリズムは、プロトコルをフレームの迅速な再伝送からバックオフさせるためにも用いられている。例えば、ラウンドトリップタイムは、再伝送無しで成功裏にピア間を伝送されたフレームそれぞれに対して計測することが可能である。この時間値は平均されて、再伝送タイムアウト値（ｒｅｔｒａｎｓｍｉｓｓｉｏｎ　ｔｉｍｅｏｕｔ　ｖａｌｕｅ）の基準となる。フレームが送られるごとに、それぞれのフレームに対してタイムアウトが設定される。あるフレームが実際に伝送されたにもかかわらず、該フレームに関する確認応答が受信されない場合、該フレームは再伝送される。このときタイムアウト値は上昇し、次の再伝送時間の基準となる。この再伝送タイムアウトには上限値および下限

値が設定されているので、その値は適切な範囲に収まるようになっている。

【０１２５】

また、インターネット・モビリティ・プロトコルでは、送信路と受信路とが別に扱われている。この手法は、本質的に非対称のチャネルにおいて特に有用である。ヒステリシスに基づいて、インターネット・モビリティ・プロトコルは自動的に、フレームサイズ（フラグメント化閾値）、保留中のフレーム数、再伝送時間、遅延承認時間などのパラメータを調整して、ネットワークを通じて送られる複製データの量を減少させる。

【０１２６】

インターネット・モビリティ・プロトコルによって、ノードが様々なネットワークの違った接続ポイントに移動することが可能になるため、下層のネットワークの特性（例えばフレームサイズ）が途中で変わってしまうことがあるが、この移動の結果、あるネットワークで伝送待ちだったフレームが、モバイル装置が現在接続している新しい媒体にはフィットしないこともありうる。このことに、フラグメント化が全てのネットワーク・インフラでサポートされているわけではないことを合わせて考慮して、フラグメント化はインターネット・モビリティ・プロトコルのレベルにおいて扱われる。それぞれのフレームが伝送される前に、インターネット・モビリティ・プロトコルにより、フレームが現時点でのフラグメント化閾値を越えているか否かが判断される。ちなみに、パフォーマンス向上の見地から、この値は現時点での最大伝送ユニットよりも少なくてよい（大きいフレームよりも小さいフレームの方が最終目的地まで到達する可能性が高いため）。プロトコルのオーバーヘッドを増加させることと、再伝送回数の増加とのトレードオフは、インターネット・モビリティ・プロトコルによって吟味され、全体的に再伝送を減少させるため、フレームサイズが減少させられることもある。あるフレームがフィットしていれば、該フレームは分割されずに伝送される。していなければ、該フレームはその接続で許容される最大のサイズにまで分割される。フレームが再伝送される場合、該フレームは再評価を受け、最大伝送ユニットが減少していれば、再度フラグメント化される（あるいは、もし最大伝送ユニットが増加していれば、該フレームはフラグメント化されない一個のフレームとして再送信されることもありうる）。

【０１２７】

プロトコルそのものは直交に設計されており、どちらの側からでもピアとの接続を確立、切断することができるようになっている。しかし、ある場合においては、実行される場所によってプロトコル・エンジンにおける些少な動作上の差異がいくつか存在し、特定の休止検出や接続寿命タイムアウトは片方の側からしか実行できないこともある。運営上必要な操作を可能にするため、モビリティ管理サーバ１０２上で動作するインターネット・モビリティ・プロトコル・エンジンは、休止期間の状況を把握している。モバイル端末システム１０４からの作業が何もないまま一定の期間が経過した場合、モビリティ管理サーバ１０２はセッションを終了させることができる。また、管理者が、ある特定の接続を確立するのに要する総所要時間を制限する、あるいは日時によってアクセスを拒否することが必要なときがあるが、この場合も、一例としてこうしたポリシータイマーはモビリティ管理サーバ１０２側からのみ操作できるようにしてもよい。

【０１２８】

一例として、インターネット・モビリティ・プロトコルを提供するソフトウェアは、ウィンドウズ（登録商標）ＮＴ、９ｘ、ＣＥ環境で、プラットフォームに合わせた修正を必要とすることなくコンパイルされ、動作できるものである。これを実現するために、インターネット・モビリティ・プロトコルでは、インターネット・モビリティ・プロトコル・フレームの送受信に、ネットワーク抽象層（ｎｅｔｗｏｒｋ ａｂｓｔｒａｃｔｉｏｎ ｌａｙｅｒ； ＮＡＬ）のサービスが採用されている。メモリ管理、キューおよびリスト管理、イベントロギング、警報システム、電力管理、セキュリティなどの他の標準的なユーティリティ機能も使われる。いくつかのランタイム・パラメータについては、エンジンがモバイル端末システム１０４に備えられているか、あるいはモビリティ管理サーバ１０２側に備えられているかによって変更される。これについて、数例を以下に示す。

【０１２９】
・特定のタイムアウトは、モビリティ管理サーバ１０２からのみ呼び出し可能
・フレームの方向は、エコー検出用のフレームヘッダそれぞれにて示される
・モバイル端末システム１０４の設定により、着信接続（ｉｎｂｏｕｎｄ　ｃｏｎｎｅｃｔｉｏｎ）の拒否が可能
・警報はモビリティ管理サーバ１０２にのみ通知される
・電力管理はモバイル端末システム１０４ではイネーブルにされるが、モビリティ管理サーバ１０２では必ずしも必要でない
インターネット・モビリティ・プロトコルのインターフェイスは、Ｃコーラブルでプラットフォームに依存しない標準的なＡＰＩ機能をほんの少数有し、作業（上記の標準的なユーティリティ機能以外）を予定する、ＯＳ特化型の単一の機能を必要とする構成でもよい。ローカルクライアントとの通信は、定義された作業オブジェクト（作業要求）の利用によって可能になる。作業要素それぞれの完了通知は、作業オブジェクトの一部として特定されたオプションの完了コールバックルーチンを通じて、要求エンティティに通知を行うことによって、効果的に達成することができる。

【０１３０】
インターネット・モビリティ・プロトコル・エンジン自体はキューベースのものである。ローカルクライアントからの作業要素は、ＦＩＦＯオーダに従ってグローバル作業キューに追加されるが、これは、ローカルクライアントが、「ＰｒｏｔｏｃｏｌＲｅｑｕｅｓｔｗｏｒｋ（）」などの標準的なインターネット・モビリティ・プロトコル機能を呼び出すことによってなされる。続いて、インターネット・モビリティ・プロトコル内のスケジューリング機能が該作業を除去し、適切な機能に対してディスパッチする。待機機能とスケジューリング機能を組み合わせることでＯＳのアーキテクチャ間の差を感じさせなくするので、プロトコル・エンジンが、スレッドベースのシステム（例えばウィンドウズ（登録商標）ＮＴ）や同期式のシステム（例えばマイクロソフト・ウィンドウズ（登録商標）９ｘやウィンドウズ（登録商標）ＣＥ）で動作することが可能になる。優先度に関するスキームを待機機能の上にオーバーレイすることができるので、（下の層がサポートしていれば）サービス品質を保証することができる。

【０１３１】
ネットワークの視点から見ると、インターネット・モビリティ・プロトコルは、データのコピーや移動を減らすべく、分散－集結技術（ｓｃａｔｔｅｒ－ｇａｔｈｅｒ　ｔｅｃｈｎｉｑｕｅｓ）を利用している。伝送はそれぞれフラグメントのリストとしてＮＡＬに送られ、ネットワーク層のトランスポートによって融合される。トランスポート・プロトコル自体が分散－集結技術をサポートしている場合、フラグメントリストは上記トランスポートを通じて転送され、媒体アクセス層ドライバあるいはハードウェアによってアセンブルされる。さらに、上記技術を拡張して、プロトコルスタックのどのレベルにおいても、いかなるプロトコルラッパーの挿入や消去を行うことも可能である。フレームの受信は、ＮＡＬ層により、ＮＡＬ登録処理（ＮＡＬ　ｒｅｇｉｓｔｒａｔｉｏｎ　ｐｒｏｃｅｓｓ）時に指定された特定の入口点においてインターネット・モビリティ・プロトコルをコールバックすることで通知される。

【０１３２】
（インターネット・モビリティ・プロトコル・エンジンのエントリポイントの例）
例示の実施形態におけるインターネット・モビリティ・プロトコルは、該プロトコルのスタートアップおよびシャットダウン行動を管理する、４つの共通エントリポイントを有している。以下に示されるのがこれらのプロシージャである。

【０１３３】
１．Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＣｒｅａｔｅ（）
２．Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｕｎ（）
３．Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＨａｌｔ（）
４．Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＵｎｌｏａｄ（）

（Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＣｒｅａｔｅ（）の例）
Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＣｒｅａｔｅ（）機能は、ブートサブシステムによって呼び出され、インターネット・モビリティ・プロトコルを初期化する。この第一フェイズ中に、作業の処理を開始するのに必要なリソース全てが所得され、初期化されなければならない。該フェイズの終了時、エンジンは、システムの他の層からの作業をすぐに受け入れ可能な状態である必要がある。このとき、インターネット・モビリティ・プロトコルはグローバル設定テーブルを初期化するが、このために、インターネット・モビリティ・プロトコルは、設定マネージャ２２８のサービスを利用して該テーブルを設定（ｐｏｐｕｌａｔｅ）する。

【０１３４】

次に、インターネット・モビリティ・プロトコルは、サスペンドおよびレジューム通知機能をＡＰＭハンドラに登録する。一例ではこれらの機能はモバイル端末システム１０４の側からのみ呼び出し可能であるが、他の例として、動作中にモビリティ管理サーバ１０２がサスペンドできるようにするのが望ましいこともある。続いて、グローバル作業キュー、グローバルＮＡＬポータルリストなど他の作業用記憶域がメモリプールから割り当てられる。

【０１３５】

必要なランタイムメモリの最大量を制限し、インターネット・モビリティ・プロトコルのハンドルの固有性を保証するため、インターネット・モビリティ・プロトコルは、ハンドル生成用の２段配列構成（２－ｔｉｅｒ　ａｒｒａｙ　ｓｃｈｅｍｅ）を採用している。グローバル接続配列テーブルのサイズは、システムが設定する同時接続の最大数によって決まり、この時点で割り当てられる。全てのグローバル記憶域が割り当てられ、初期化されると、グローバル・インターネット・モビリティ・プロトコルの状態が、＿ＳＴＡＴＥ＿ＩＮＩＴＩＡＬＩＺＥ＿となる。

【０１３６】

（Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｕｎ（）の例）
Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｕｎ（）機能は、全てのサブシステムが初期化された後で呼び出され、インターネット・モビリティ・プロトコルのサブシステムに、待機中の作業をどれでも処理し始めてよいことを通知する。これが、一般的な動作状況におけるインターネット・モビリティ・プロトコル・エンジンの通常の状態である。該エンジンを動作状態にする前に、いくつかの第二パス初期化ステップがこの時点で実行される。

【０１３７】

インターネット・モビリティ・プロトコルは、ネットワーク通信が、任意のインターフェイスのどれを通じてでも実行されるようにする。初期化ステップの間に、インターネット・モビリティ・プロトコルーＮＡＬ間のインターフェイス用の記憶域が割り当てられるので、インターネット・モビリティ・プロトコルはグローバルポータルリスト中を横断して、ＮＡＬの全てのリスナをスタートさせることができる。一例として、この動作は２つのステップを含んだ以下の処理のようになる。

【０１３８】

・インターネット・モビリティ・プロトコルは、ＮＡＬ層に、初期化中に供給される設定に基づいたポータルを結びつけて（ｂｉｎｄ）オープンし、
・インターネット・モビリティ・プロトコルは、次に、Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲＣＶＦＲＯＭＣＢコールバックを登録して、受信したフレームの処理を始める準備ができたことを、ＮＡＬ層に通知する。

【０１３９】

・引き続き、ローカル持続識別子（ｌｏｃａｌ　ｐｅｒｓｉｓｔｅｎｔ　ｉｄｅｎｔｉｆｉｅｒ；　ＰＩＤ）が初期化される。グローバル・インターネット・モビリティ・プロトコルの状態は、＿ＳＴＡＴＥ＿ＲＵＮ＿に変わる。

【０１４０】

（Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＨａｌｔ（）の例）
Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＨａｌｔ（）機能は、システムがシャットダウンしたことをエンジンに警告するために呼びだされる。動作中に得られたリソースは、この機能から戻る前に全て開放される。インターネット・モビリティ・プロトコルの全てのセッションは、管理用に設定された理由コードによって異常終了する。エンジンが＿ＳＴＡＴＥ＿ＨＡＬＴＥＤ＿状態になると、それ以降、他の層からの作業は受け付けられず、他の層に通知されることも無い。

【０１４１】

（Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＵｎｌｏａｄ（）の例）
Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＵｎｌｏａｄ（）機能は、シャットダウン処理の第二フェイズである。これが、復帰以前に、割り当てられたシステムリソースをエンジンが解放する最後の機会となる。エンジンがこの機能から戻ると、システム自体が停止するので、これ以上の作業は実行されない。

【０１４２】

（インターネット・モビリティ・プロトコル・ハンドルの例）
少なくともいくつかの例においては、メモリ（インターネット・モビリティ・プロトコル状態情報を記憶）のアドレスを、インターネット・モビリティ・プロトコルの接続を記述するトークンとして用いるだけでは不十分である。これは主に、短い期間のうちに、ある接続が終了して別の新しい接続が開始される可能性のためである。メモリアロケータが、別の接続に前のものと同じアドレスを再配分してしまう可能性が高く、この値が前の接続と新しい接続との両方を示すことになってしまう。元々のピアが、（電源が切られていたり、サスペンドしていたり、圏外にあったりなどで）セッションの終了を認識しなかった場合、前のセッションのフレームを新しいセッションの方に送ってしまうこともあり得る。ＴＣＰの場合はこれが起こり、ピアのＩＰアドレスが同じ場合、新しいセッションを生成するためにリセットが行われる。これを避けるため、インターネット・モビリティ・プロトコルは製造ハンドル（ｍａｎｕｆａｃｔｕｒｅｄ　ｈａｎｄｌｅ）を採用している。該ハンドルは２列のインデックスによって形成され、独自性を保つため一回限りのものである。テーブルは以下のようなレイアウトとなっている。

【０１４３】

テーブル１：接続オブジェクトの配列を指すポインタの配列
テーブル２：インターネット・モビリティ・プロトコル制御ブロックを指す、実ポインタを含む接続オブジェクトの配列
この技術により、初期化期間に割り当てられるメモリの量が最小化される。テーブル１は、スタートアップの際にサイズが決められ、割り当てがなされる。これにより、モバイル端末システム１０４の側では、少量のメモリの割り当てが可能になる（モビリティ管理サーバ１０２側でテーブル１が割り当てを要求するメモリは、サーバが多くの接続を有しているため少し大きなものになる）。

【０１４４】

テーブル１は必要時に設定（ｐｏｐｕｌａｔｅ）される。接続要求が出されると、インターネット・モビリティ・プロトコルは、テーブル１内でテーブル２への有効なポインタを探す。エントリポイントが見つからない場合、インターネット・モビリティ・プロトコルは、新規のテーブル２に２５６個の接続オブジェクトを割り当て、テーブル２へのポインタを、テーブル１の適切なスロットに記録する。次に、プロトコル・エンジンはテーブル２を初期化し、新しく生成されたテーブルからの接続オブジェクトを割り当てて、製造ハンドルへ戻る。他のセッションが要求された場合、インターネット・モビリティ・プロトコルは再度テーブル１中をサーチして、テーブル２への有効なポインタを見つけ、該セッション用の次の接続オブジェクトを割り当てる。この作業は、以下の二つの状態のうちのどちらかになるまで続けられる。

【０１４５】

・テーブル２から全ての接続オブジェクトが無くなった場合、新たにテーブル２が割り当

てられ、初期化されて、該テーブル２を示すポインタが、テーブル１の次に利用可能なスロットに設定される。

【０１４６】

・全ての接続オブジェクトが特定のテーブル２のインスタンスに開放され、全ての要素がある特定の期間使用されなかった場合、テーブル２の上記インスタンスにおける記憶域が開放されてメモリプールに戻され、テーブル１の関連しているポインタはゼロにされて、次の接続要求が開始された際、該入口が使用可能であることを示すようになる（テーブル２の他のインスタンスで、利用可能な接続オブジェクトがない場合に限る）。

【０１４７】

二つのグローバルカウンタは、割り当てられる接続の総数を制限できるようにするために維持されている。片方のグローバルカウンタは現時点でアクティブな接続を数え、もう片方のカウンタは、割り当てられていない接続オブジェクトの数を把握する。さらに、第二のカウンタは、任意に設定された制限まで生成可能な接続オブジェクトの総数の管理も行う。新規のテーブル２が割り当てられると、該カウンタは、該新規のテーブルが表現するオブジェクトの数を把握するため、下方に調整される。反対側では、インターネット・モビリティ・プロトコルがテーブル２のインスタンスをメモリプールに開放するときに、カウンタは、開放された接続オブジェクトの数を合わせて上方修正される。

【０１４８】

（ワークフローの例）

Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｅｑｕｅｓｔＷｏｒｋ（）機能によって、作業はローカルクライアントから要求される。該作業が検査（ｖａｌｉｄａｔｅ）され、グローバル作業キューに加えられると、Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＱｕｅｕｅＥｌｉｇｉｂｌｅ（）機能が呼び出される。スレッド化された環境であれば、インターネット・モビリティ・プロトコルのワーカースレッドに信号が送られ（適合マークがつけられ）、制御は直ちに呼び出しエンティティへと戻る。同期的な環境であれば、グローバル作業キューが直ちに実行されて、要求された作業はいかなるものであっても処理される。どちらの場合でも、Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＰｒｏｃｅｓｓＷｏｒｋ（）機能が実行されることになる。これが作業処理のためのメインディスパッチ機能である。

【０１４９】

例示の実施形態では、グローバルキューからの作業をディスパッチできるのが一回に一つのスレッドのみの場合があり、再入を防ぐべくグローバルセマフォが用いられてもよい。プライベートなインターネット・モビリティ・プロトコルの作業では、Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｅｑｕｅｓｔＷｏｒｋ（）機能を使用せず、直接グローバル作業キューに作業を送ることができる。

【０１５０】

ＳＥＮＤタイプの作業オブジェクトでは、特殊な場合が想定される。信頼性の低いデータグラムのセマンティクスが保たれていることを保証するため、ＳＥＮＤタイプの作業オブジェクトをそれぞれ、有効期限あるいは再試行カウントを伴って待機させることができる。作業は該有効期限に基づいて寿命を算定される。このような所定のタイムアウトが生じた場合、作業オブジェクトは接続特定キューから除かれ、エラー状態で完了する。ＳＥＮＤオブジェクトが既にデータパスに融合されている場合、プロトコルは、再試行カウントが設定されたどのＳＥＮＤオブジェクトでも、除去を許可する。再試行カウントを超過した場合、該オブジェクトは、特定のフレームを形成する要素のリストから外され、適当なエラー状態を示しつつリクエスタへ戻される。

【０１５１】

（接続スタートアップの例）

インターネット・モビリティ・プロトコルは、ピア間の接続を確立する非常に効率的なメカニズムを有している。接続の確認は、ピア間で最低３フレームをやり取りすることでなされる。開始側は、そのピアにＩＭＰ　ＳＹＮＣフレームを送って接続確立が要求されて

いることを警告する。受け入れ側は、ＩＭＰ　ＥＳＴＡＢＬＩＳＨフレームを送って接続受け入れを確認するか、あるいはＩＭＰ　ＡＢＯＲＴフレームを送ることで、接続要求が拒否されたことをピアに警告する。ユーザに接続拒否の理由を理解しやすくするため、理由および状態コードが、ＩＭＰ　ＡＢＯＲＴフレームに含まれる形で送られる。接続が許可されると、確認応答フレーム（プロトコル・データ・ユニットおよび制御データがふくまれていることもある）が送られて、受け入れ側に転送され、確立フレームの受信が確認される。

【０１５２】

ネットワークのトラフィックを最小化するため、プロトコルは、ユーザおよび制御データを、接続スタートアップ時の最初のハンドシェーク機構に含まれるようにすることができる。この構成は、セキュリティが保障されていない環境や、同じデータパスでセキュリティ用の認証が二重に行われたり、暗号化処理が同じデータパスで行われたりすることによるパフォーマンスの低下を避けるよう、インターネット・モビリティ・プロトコルが調整されているというような、セキュリティが下層で扱われている環境において使用される。

【０１５３】

（データ転送の例）

インターネット・モビリティ・プロトコルは、フレームがネットワークに配信されたことを、ＮＡＬからの通知によって検知する。インターネット・モビリティ・プロトコルは、当該ネットワークリンクが絶えずフロー制御されているか否かを上記メトリクスによって判定するので、当初の要求が完了されるまで同じフレームを再伝送することはない。しかし、ネットワークドライバによっては、フレームの伝送について、誤って、該フレームをネットワークに送る前に配信を示唆してしまうものもある。セマフォの利用により、インターネット・モビリティ・プロトコル層はこれを検知し、当初の要求についてＮＡＬが返答するまでは、他のデータグラムを送信しない。

【０１５４】

フレームがインターネット・モビリティ・プロトコルに受信されると、該フレームは迅速に検査されて、適切な接続キューに加えられる。該フレームが、インターネット・モビリティ・プロトコルがその最終目的地を識別するのに足る情報を有していない場合、該フレームは、それを受信したインターネット・モビリティ・プロトコル・ソケット・キューに加えられ、該ソケット・キューは、続く処理のためにグローバル作業キューに加えられる。この最初の逆多重化により、受信した作業は、処理オーバーヘッドが限定された状態で、迅速に拡散させられる。

【０１５５】

（黙認（ａｃｑｕｉｅｓｃｉｎｇ）の例）

再伝送時のネットワーク帯域幅およびモビリティ管理サーバ１０２の処理に要する電力の最小化のため、プロトコルは、モビリティ管理サーバ１０２が接続を「黙認」できるようにする。ユーザによる設定が可能な期間の後、モビリティ管理サーバ１０２は、対応するモバイル端末システム１０４から通知が無ければ、特定の接続へのフレームの再伝送を停止する。この時、モビリティ管理サーバ１０２は、モバイル端末システム１０４が通信できない状態（圏外、サスペンドなど）であると仮定し、接続を休止状態にする。該接続へのこの後の作業は全て、後の配信のために記憶される。該接続は、以下の状況のいずれかが満たされない限り、同じ状態にとどまり続ける。

【０１５６】

・モビリティ管理サーバ１０２がモバイル端末システム１０４からフレームを受信し、接続を本来の状態に戻す、
・接続寿命タイムアウトの期限が切れる、
・休止タイムアウトの期限が切れる、または
・システム管理者によって接続が中止される
モビリティ管理サーバ１０２が、モバイル端末システム１０４からフレームを受信し、中断された所から接続が再開された場合、該接続で待機していた作業は全て転送され、状態

が再同期化される。これ以外の場合、再接続がなされると、モバイル端末システム１０４には以前の接続の終了が通知され、モバイル端末システム１０４に送られることを待機していた作業は破棄される。

【０１５７】

（接続および送信要求の例）

図１０Ａ～１０Ｃは、インターネット・モビリティ・エンジン２４４によって生成される接続および送信要求ロジックを例示する、フローチャートを形成している。ＲＰＣエンジン２４０からの命令を受信すると、インターネット・モビリティ・プロトコル・エンジン２４４は、該命令が「接続」要求かどうかを判定する（判定ブロック６０２）。該命令が「接続」要求であれば、エンジン２４４は、接続リソースの割り当てが可能かどうかを判断する（判定ブロック６０３）。十分な接続リソースが割り当てられない場合（判定ブロック６０３で「ノー」である場合）、エンジン２４４はエラーを宣言して（ブロック６０３ａ）応答する。接続リソースの割り当てが可能であれば、エンジン２４４は状態設定処理を行って、接続要求の処理に備える（ブロック６０３ｂ）。

【０１５８】

接続その他の要求に対し、エンジン２４４は該接続あるいは送信要求を待機させ、呼び出しを実行しているアプリケーションに応答する前に、グローバルイベントに通知する（ブロック６０４）。

【０１５９】

インターネット・モビリティ・プロトコルのグローバル要求キューからの接続あるいは送信要求をディスパッチするために、エンジン２４４はまず、保留されている作業の有無を判定する（判定ブロック６０５）。保留されている作業が無ければ（判定ブロック６０５で「ノー」である場合）、エンジン２４４は、図１０Ｃのブロック６２５に移って、接続用の作業を待機させるため、アプリケーションからの応答を待つ（ブロック６０５Ａ）。保留されている作業があれば（判定ブロック６０５で「イエス」の場合）、エンジン２４４は現在の状態が確立されているか否かを判定する（ブロック６０６）。状態が確立されている場合（判定ブロック６０６で「イエス」の場合）、エンジン２４４は状態確立への移行のためのステップを飛ばして、図１０Ｂの判定ブロック６１５に移る（ブロック６０６ａ）。状態が確立されていない場合、エンジン２４４は状態確立のための一連のステップを実行する必要がある（判定ブロック６０６で「ノー」である場合）。

【０１６０】

状態確立に移行するために、エンジン２４４はまず、そのピアのアドレスが既知かどうかを判定する（判定ブロック６０７）。既知でなければ、エンジン２４４は作業をさらに待機させ、図１０Ｃのブロック６２５に移って、ピアのアドレスが来るのを待つ（ブロック６０７ａ）。ピアのアドレスが既知の場合（判定ブロック６０７で「イエス」の場合）、エンジン２４４は次に、必要なセキュリティコンテキストが取得されているかどうかを判定する（判定ブロック６０８）。取得されていない場合、エンジン２４４は作業をさらに待機させ、図１０Ｃのブロック６２５に移って、セキュリティコンテキストが来るのを待つ必要がある（ブロック６０７ａ）。セキュリティコンテキストが既に取得されている場合（判定ブロック６０８で「イエス」の場合）、エンジン２４４は「状態保留」状態を宣言して（ブロック６０８ｂ）、インターネット・モビリティ・プロトコルの同期フレームを送信し（ブロック６０９）、再伝送タイマーをスタートさせる（ブロック６１０）。エンジン２４４は、対応する確立フレームが受信されたかどうかを判定する（ブロック６１１）。受信されていない場合（判定ブロック６１１で「ノー」の場合）、エンジン２４４は、再伝送時間の期限が来たかどうかを判定する（判定ブロック６１２）。再伝送時間の期限が来ていない場合（判定ブロック６１２で「ノー」の場合）、エンジン２４４は待機し、続いてステップ６２５に移ってもよい（ブロック６１３）。最終的に、確立フレームが受信されず（ブロック６１１で判定）、再伝送時間の期限が全て切れた（判定ブロック６１４）場合、接続は中止されてもよい（ブロック６１４ａ）。結局確立フレームが受信された場合（判定ブロック６１１で「イエス」の場合）、エンジン２４４は「状態確立」

状態を宣言する（ブロック６１１ａ）。

【０１６１】

状態確立がなされると、エンジン２４４は新規の接続の認証がなされているかどうかを判定する（判定ブロック６１５）。なされていない場合、エンジン２４４は待機し、ステップ６２５へと移ってよい（ブロック６１６）。接続の認証がなされている場合（判定ブロック６１５で「イエス」の場合）、エンジン２４４は認証が成功したかどうかを判定する（判定ブロック６１７）。成功していない場合（判定ブロック６１７で「ノー」の場合）接続は中止される（ブロック６１４ａ）。成功している場合、エンジン２４４は、ピア伝送ウィンドウがフルになっているかどうかを判定する（判定ブロック６１８）。フルになっていれば（判定ブロック６１８で「イエス」の場合）、エンジン２４４は確認応答を待ち、ステップ６２５に進む（判定ブロック６１９）。ウィンドウがフルになっていなければ（判定ブロック６１８で「ノー」の場合）、エンジン２４４はインターネット・モビリティ・プロトコル・データフレームを生成し（ブロック６２０）、送信する（ブロック６２１）。次に、エンジン２４４は再伝送タイマーがスタートしているかどうかを判定する（判定ブロック６２２）。していない場合、エンジン２４４は再伝送タイマーをスタートさせる（ブロック６２３）。エンジン２４４は、それ以上送信するデータが無くなるまで（判定ブロック６２４で判断される）、ブロック６１８～６２３を繰り返す。そして、エンジン２４４はスリープモードに入ってさらに作業が来るのを待ち、グローバルディスパッチャに戻る（ブロック６２５）。

【０１６２】

（中止の例）

図１１は、接続を中止するためにインターネット・モビリティ・プロトコル・エンジン２４４が実行するステップを例示したフローチャートである。「接続中止」要求を受けて（ブロック６２６）、該エンジンは該要求をグローバル作業キューに加え、呼び出しを行っているアプリケーションに返答する（ブロック６２６ａ）。中止要求は最終的に、実行のため、インターネット・モビリティ・プロトコルの処理グローバル作業キューからディスパッチされる（ブロック６２７）。エンジン２４４は中止要求を調べて、該要求が緊急のものであるか、あるいは余裕のあるものであるかを判定する（判定ブロック６２８）。緊急のものの場合（判定ブロック６２８で「中止」の場合）、エンジン２４４は直ちに接続を中止する（ブロック６２９）。余裕があるものであれば（判定ブロック６２８で「余裕」の場合）、エンジン２４４は「状態クローズ（ｓｔａｔｅ　ｃｌｏｓｅ）」状態を宣言し（ブロック６２８ａ）、インターネット・モビリティ・プロトコルの「モーティス（Ｍｏｒｔｉｓ）」フレームを送信し（ブロック６３０）、ピアに接続がクローズされることを示唆する。次に、エンジン２４４は「モーティス」状態を宣言し（ブロック６３０ａ）、再伝送タイマーをスタートさせる（ブロック６３１）。エンジン２４４は、ピアから「ポスト・モーテム（ｐｏｓｔ　ｍｏｒｔｅｍ）」フレームが応答されたかどうかを判定する（判定ブロック６３２）。応答されていなければ（判定ブロック６３２で「ノー」の場合）、エンジン２４４は、再伝送タイマーが時間切れになっているか否かを判定する（判定ブロック６３３）。まだ時間切れになっていない場合（判定ブロック６３３で「ノー」の場合）、エンジン２４４は待機し、ステップ６３７へ進む（ブロック６３４）。再伝送タイマーが時間切れになっている場合（判定ブロック６３３で「イエス」の場合）、エンジン２４４は総再伝送時間の期限が切れているかどうかを判定する（判定ブロック６３５）。まだ切れていない場合（判定ブロック６３５で「ノー」の場合）、ブロック６３０に戻り、モーティスフレームを再送する。すでに総再伝送時間の期限が切れている場合（判定ブロック６３５で「イエス」の場合）、エンジン２４４は直ちに接続を中止する（ブロック６３５ａ）。

【０１６３】

「ポスト・モーテム」応答フレームがピアから受信されると（判定ブロック６３２で「イエス」）、エンジン２４４は「ポスト・モーテム」状態を宣言して（ブロック６３２ａ）、接続リソースを開放し（ブロック６３６）、スリープ状態に戻ってさらに作業を待つ（

ブロック６３７）。

【０１６４】

（再伝送の例）

図１２は、インターネット・モビリティ・プロトコル・エンジン２４４が実行する「再伝送」イベントロジックの例を示している。再伝送タイマーが時間切れを迎える（ブロック６５０）と、エンジン２４４は、保留中のフレームがあるかどうかを判定する（判定ブロック６５１）。保留中のフレームがなければ（判定ブロック６５１で「ノー」の場合）、エンジン２４４はタイマーを破棄して（ブロック６５２）スリープ状態に戻る（ブロック６６０）。逆に保留中のフレームがあれば（判定ブロック６５１で「イエス」の場合）、エンジン２４４は、再伝送期間全体（ｅｎｔｉｒｅ）が終了したかどうかを判定する（判定ブロック６５３）。まだ終了していなければ（判定ブロック６５３で「ノー」の場合）、処理は時間差のためにスリープ状態に戻る（ブロック６５４）。既に終了していれば（判定ブロック６５３で「イエス」の場合）、エンジン２４４は、総ての（ｔｏｔａｌ）再伝送期間が終了したかどうかを判定する（判定ブロック６５５）。既に終了しており（判定ブロック６５５で「イエス」）、このイベントがモビリティ管理サーバのエンジン２４４’（モバイル端末システムのエンジン２４４に対応）で実行されている場合、休眠状態が宣言される（判定ブロック６５６、ブロック６５６ａ）。同じ状態で、モバイル端末システム１０４において実行されるインターネット・モビリティ・プロトコル・エンジン２４４は、接続を中止する（ブロック６５６ｂ）。

【０１６５】

総ての再伝送期間がまだ終了していない場合（判定ブロック６５５で「ノー」の場合）、エンジン２４４はフレームを再処理して時間切れのデータを除去し（ブロック６５７）、それを再転送し（ブロック６５８）、再転送タイマーをそのまま再始動させる（ブロック６５９）。そして処理は休眠状態に入り（ブロック６６０）、次のイベントを待つ。

【０１６６】

（インターネット・モビリティ・プロトコルのＰＤＵの期限切れの例）

図１２のブロック６５７において、要求を出している上層のインターフェイスが、結合しているピアに送られるプロトコル・データ・ユニット（つまりＳＥＮＤ作業要求）５０６の期限切れに関するタイムアウトあるいは再試行カウントを特定することが可能になる。この機能により、インターネット・モビリティ・プロトコル・エンジン２４４は、信頼性の低いデータのセマンティクス維持、再伝送されたデータからの信頼性の低いデータの除去など、他の機能も果たせるようになる。上層からのＰＤＵ（プロトコル・データ・ユニット）それぞれは、最終的にインターネット・モビリティ・プロトコル・エンジン２４４によってコーリシングされる、それぞれの要素に対する有効期限タイムアウト（ｖａｌｉｄｉｔｙ　ｔｉｍｅｏｕｔ）および／または再試行カウントを、特定することができる。有効期限タイムアウトおよび／または再試行カウント（アプリケーションによってはユーザによって特定可能）は、エンジン２４４による再伝送の前に、どのＰＤＵ５０６が、転送されずにフレームから除去されるべきかを決定するために使われる。

【０１６７】

ＰＤＵ５０６に関わる有効期限タイムアウトは、ＰＤＵそれぞれが伝送に際して考慮すべき、相対期間（ｒｅｌａｔｉｖｅ　ｔｉｍｅ　ｐｅｒｉｏｄ）を特定する。実行依頼の間、Ｉｎｔｅｒｎｅｔ　Ｍｏｂｉｌｉｔｙ　ＰｒｏｔｏｃｏｌＲｅｑｕｅｓｔＷｏｒｋ機能によって有効期限の値（ｅｘｐｉｒｙ　ｔｉｍｅｏｕｔ　ｖａｌｕｅ）がチェックされる。該値が０で無い場合、時期タイマー（ａｇｅ　ｔｉｍｅｒ）は初期化される。次に、要求されたデータは、結合しているピアに送られる他のデータと同じキューに加えられる。あるＰＤＵ５０６が、有効期限パラメータ（ｖａｌｉｄｉｔｙ　ｐｅｒｉｏｄ　ｐａｒａｍｅｔｅｒ）が特定する期間よりも長くキューにある場合、該キューを処理する次イベントの間に、有効期限の切れた該（全ての）ＰＤＵは除去され、フレームの再伝送時に再伝送されるのではなく、ローカルに「タイムアウトによる失敗（ｔｉｍｅｏｕｔ　ｆａｉｌｕｒｅ）」のステータスコードをもって完了する。このアルゴリズムにより、ピアへの伝

送のために待機している信頼性の低いデータが古くなる（ｇｒｏｗ　ｓｔａｌｅ）こと、および／または際限なくシステムリソースを消費することを防ぐ。

【０１６８】

図１２Ａの例において、少なくとも３つの独立したＰＤＵ５０６が、続く処理のためにインターネット・モビリティ・プロトコル・エンジン２４４でキューに加わっている。ＰＤＵ５０６（１）は有効期限を持たず、要求に対するタイムアウトが無い。ＰＤＵ５０６（２）は、２秒の有効期間を持ち、時系列的にＰＤＵ５０６（１）の後に並んでいる。ＰＤＵ５０６（ｎ）は、２．５秒の有効期間を持ち、時系列的にＰＤＵ５０６（２）の後に並んでいる。ＰＤＵ５０６（ｎ）の待機が、キューの処理をさせ、ＰＤＵ５０６（２）の有効期間を経過させる最初のイベントであるため、ＰＤＵ５０６（２）は作業キューから外され、ローカルに完了されて、ＰＤＵ５０６（ｎ）がリストに加えられる。有効期間がＰＤＵ５０６（ｎ）に設定されている場合、ここまでの一連のイベントが繰り返される。作業キューを操作するいかなるイベント（キューに加える、キューから外す等）によっても、古くなったＰＤＵが除去され、完了される。

【０１６９】

上記のように、ＰＤＵ５０６はインターネット・モビリティ・プロトコル・エンジン２４４の伝送ロジックによって融合され、単一のデータストリームにフォーマットされる。独立した作業要素それぞれは、有効期限タイムアウトによって期限切れになっていなければ、集められてインターネット・モビリティ・プロトコル・データフレームを形成する。インターネット・モビリティ・プロトコル・エンジン２４４は、最終的にこれらのＰＤＵ５０６をピアに送り、関係するフレームをフレーム保留リスト（Ｆｒａｍｅｓ－Ｏｕｔｓｔａｎｄｉｎｇ　ｌｉｓｔ）に加える。ピアが一定期間内に該フレームを認識しなかった場合（図１２の再伝送アルゴリズムを参照）、フレームは再伝送され、失われたかあるいは破損したパケットを回復する。再伝送の直前、フレームを形成するＰＤＵリストは、再試行カウントを伴った要求が待機しているかどうかを判定すべく反復される。該再試行カウントが０ではなく、０に向かって減少してゆく場合、ＰＤＵ５０６はリストから外され、フレームヘッダはデータの消去を示すように修正される。このように、古くなったデータ、低信頼のデータ、あるいは独自の再伝送ポリシーを持つアプリケーションは、エンジン２４４’の再伝送アルゴリズムによって負担をかけられることがない。

【０１７０】

図１２Ｂの例においても、少なくとも３つの独立したＰＤＵ５０６が、続く処理のためにインターネット・モビリティ・プロトコル・エンジン２４４でキューに加わっている。ＰＤＵ５０６（１）は再試行カウント無しでキューに加わっているが、これは、継続的な再伝送の試行と、保証された配送レベルのサービスとを示している。ＰＤＵ５０２（２）は再試行カウントが１でキューに加わり、時系列的にＰＤＵ５０６（１）の後に並ぶ。ＰＤＵ５０６（ｎ）は、ある程度の時間が経過してからＰＤＵ５０２（２）より後ろに加わる。この時点で、いくつかの外部イベント（例えば上層融合・タイマー）が、エンジン２４４’に、インターネット・モビリティ・プロトコル・データフレーム５００を生成するための作業キューから十分なＰＤＵ５０６を集めて新しいフレームを生成するための論理を送信させる。フレームヘッダ５０３が求められ、これに、フレームの最初の伝送であることを示すための、再試行ＩＤとして０が付けられる。続いてフレームは、ネットワークへの後の伝送用に、ＮＡＬ層へと送られる。このときに再伝送タイマーがスタートするが、これは、当該のフレームがペイロードを含んでいるためである。説明を容易にするため、再伝送タイマーが期限を迎える前に、様々な理由でピアからの確認応答が受け取られなかったと仮定すると、エンジン２４４の再伝送ロジックにより、当該するフレーム５００が、ネットワークへ再伝送されるのに的確であるか否かが判断される。フレームをＮＡＬ層に再送する前に、エンジン２４４’の再伝送ロジックは、関連しているＰＤＵ５０６のリストを反復する。ＰＤＵ５０６（２）それぞれの再試行カウントが調べられ、０でなければ、該カウントは減少させられる。ＰＤＵ５０６（２）の再試行カウントを減少させる処理により、該カウントは０になる。ＰＤＵ５０６（２）の再試行カウントが０になったこ

とにより、ＰＤＵ５０６（２）はリストから外され、「再試行失敗（ｒｅｔｒｙ　ｆａｉｌｕｒｅ）」のステータスをもってローカルに完了する。続いて、ＰＤＵ５０６（２）のデータが無いことを示すため、フレームヘッダ５０３のサイズが調整される。この処理は、残っている全てのＰＤＵについて繰り返される。フレーム５００全体が、「編集後」フレーム５００’の作成のために再処理されると、ヘッダの再試行ＩＤが増加させられて、生成されたデータグラムが、続く（再）伝送に向けてＮＡＬ層に送られる。

【０１７１】

（受信の例）

図１３Ａ～１３Ｄは、「受信」イベント受信に伴ってインターネット・モビリティ・プロトコル・エンジン２４４が実行するステップを例示する、フローチャートを形成している。このような受信イベントは、インターネット・モビリティ・プロトコル・エンジン・フレームがネットワーク１０８から送られた際に生成される。この受信イベントを受けて、エンジン２４４は該イベントを事前検査（ｐｒｅ－ｖａｌｉｄａｔｅ）し（ブロック６７０）、インターネット・モビリティ・プロトコル・エンジン・フレームである可能性があるか否かを判定する（判定ブロック６７１）。エンジンが可能性無しと判断した場合（判定ブロック６７１で「ノー」の場合）、該フレームは破棄される（ブロック６７２）。可能性ありという判断の場合（判定ブロック６７１で「イエス」の場合）、エンジン２４４は、受信したフレームと関連する接続があるかどうかを判定する（判定フレーム６７３）。受信したフレームと関連する接続がある場合（判定ブロック６７３で「イエス」の場合）、エンジン２４４は作業を接続受信キューに加え（ブロック６７４）、該接続に受信に適格であるとマークを付け（ブロック６７５）、該接続をグローバル作業キューに加える（ブロック６７６）。まだどの接続も受信したフレームと関連していない場合（判定ブロック６７３で「ノー」の場合）、エンジン２４４は、受信したフレームをソケット受信キューに加え（ブロック６７７）、該ソケット受信キューをグローバル作業キューに加える（ブロック６７８）。どちらの場合でも、エンジン２４４はグローバル作業イベントを送る（ブロック６７９）。「受信適格」イベントをグローバル作業キューからディスパッチする際（図１３Ｂ参照）、エンジン２４４はフレームをそれぞれの受信キューから外す（ブロック６８０）。インターネット・モビリティ・プロトコル・エンジン２４４がメッセージをキューから外し始められるようになる前に、複数のＩＭＰフレームを受信し、キューに加えることが可能である。エンジン２４４は、全てのフレームがキューから外れるまで作業を繰り返す（ブロック６８１、６８２）。あるフレームがキューから外れると（判定ブロック６８１で「イエス」）、エンジン２４４は受信したフレームを検査し（ブロック６８３）、該フレームが有効かどうかを判定する（判定ブロック６８４）。該受信したフレームが無効である場合、エンジン２４４は該フレームを破棄し（ブロック６８５）、受信キューから次のフレームを外す（ブロック６８０）。上記フレームが有効である場合（判定ブロック６８４で「イエス」の場合）、エンジン２４４は、該フレームが既存の接続と関連しているか否かを判定する（ブロック６８６）。していなければ（判定ブロック６８６で「ノー」の場合）、エンジン２４４は同期フレームがあるかどうかを判定する（判定ブロック６８７）。同期フレームがなければ（判定ブロック６８７で「ノー」の場合）、該フレームは破棄される（ブロック６８５）。反対に、同期フレームが受信されていれば（判定ブロック６８７で「イエス」の場合）、エンジン２４４は、図１４Ａおよび１４Ｂを参照して説明される受動接続要求により、該フレームを処理する（ブロック６８８）。

【０１７２】

上記フレームが接続と関係していれば（判定ブロック６８６で「イエス」の場合）、エンジン２４４は接続状態が依然アクティブで、まだ「ポスト・モーテム」になっていないかどうかを判定する（判定ブロック６８９）。接続が既に「ポスト・モーテム」である場合、フレームは破棄される（ブロック６８５）。そうでない場合、エンジン２４４はフレームを解析し（ブロック６９０）、該フレームが中止フレームであるかどうかを判定する（判定ブロック６９１）。該フレームが中止フレームであれば、エンジン２４４は直ちに接

続を中止する（ブロック６９１ａ）。該フレームが中止フレームでなければ（判定ブロック６９１で「イエス」の場合）、エンジン２４４は確認応答情報を処理し、全ての保留中の送信フレームを開放する（ブロック６９２）。次に、エンジン２４４は、解読の必要がある場合のため、セキュリティ用のサブシステムへと送る（ブロック６９３）。フレームがセキュリティ用のサブシステムから戻ると、エンジン２４４は全ての制御データを処理する（ブロック６９４）。続いて、エンジン２４４はフレームがアプリケーションデータを含んでいるかどうかを判定する（判定ブロック６９５）。含んでいる場合、該データはアプリケーション層においてキューに加えられる（ブロック６９６）。エンジン２４４はまた、接続が休止状態であるかどうかを判定し（ブロック６９７および６９７ａ：これは、好ましい実施形態におけるモビリティ管理サーバのエンジン２４４’にもあてはまる）、確立状態に戻す。

【０１７３】

フレームが「モーティス」フレームである可能性があれば（判定ブロック６９８で「イエス」の場合）、エンジン２４４は、アプリケーション層に「切断」を示唆し（ブロック６９９）、「モーティス」状態に入る（ブロック６９９ａ）。エンジン２４４は「ポスト・モーテム」フレームをピアに送り（ブロック７００）、「ポスト・モーテム」状態に入る（ブロック７００ａ）。続いて、エンジン２４４は接続リソースを解放し（ブロック７０１）、スリープ状態に戻って作業を待つ（ブロック７０２）。解析されたフレームが「ポスト・モーテム」フレームだった場合（判定ブロック７０３で「イエス」の場合）、ブロック７００ａ、７０１、７０２が実行される。これ以外の場合、制御はブロック６８０に戻って、次のフレームを受信キューから外す（ブロック７０４）。

【０１７４】

（受動接続の例）

図１４Ａ～１４Ｂは、「受動接続」要求に対してインターネット・モビリティ・プロトコル・エンジン２４４が実行するステップを例示する、フローチャートを形成している。まず、エンジン２４４は、当該装置に他の接続が存在するかどうかを判定する（ブロック７２０）。存在すれば（判定ブロック７２０で「イエス」の場合）、該エンジンは、それが最初の接続であるか否かを判定する（判定ブロック７２１）。ピアが、新規の接続が最初の接続であると認識している場合（判定ブロック７２１で「イエス」の場合）、エンジン２４４はそれ以前の接続を中止する（ブロック７２２）。最初の接続ではない場合（判定ブロック７２１で「ノー」の場合）、エンジン２４４は、シーケンスと接続ＩＤとが対応しているかどうかを判定する（判定ブロック７２３）。対応していなければ（判定ブロック７２３で「ノー」の場合）、制御は判定ブロック７２０へ戻る。シーケンスと接続ＩＤが対応していれば（判定ブロック７２３で「イエス」の場合）、エンジン２４４は複製フレームを破棄し（ブロック７２４）、図１３Ｂのステップ６８０に戻る（ブロック７２５）。

【０１７５】

他の接続が無ければ（判定ブロック７２０で「ノー」の場合）、エンジン２４４は、接続に接続リソースを割り当て可能か否かを判定する（判定ブロック７２６）。割り当て可能でない場合、エラーが宣言され（判定ブロック７２６で「ノー」の場合、ブロック７２７）、接続が中止される（ブロック７２８）。接続リソースを割り当て可能な場合（判定ブロック７２６で「イエス」の場合）、エンジン２４４は「設定」状態を宣言し（ブロック７２６ａ）、接続についてのセキュリティコンテキストを取得する（ブロック７３０）。十分なセキュリティコンテキストの取得ができなかった場合（判定ブロック７３１で「ノー」の場合）、接続は中止される（ブロック７２８）。取得できた場合、エンジン２４４は確立フレームを送り（ブロック７３２）、接続が「確立」状態になったことを宣言する（ブロック７３２ａ）。続いて、エンジン２４４は再送部（ｒｅｔｒａｎｓｍｉｔｔｅｒ）を始動させ（ブロック７３３）、完了に向けて認証処理を待つ（ブロック７３４）。最後に、エンジン２４４は装置とユーザとが両方認証済みか否かを判定する（ブロック７３５）。装置とユーザのどちらかが認証されていない場合、接続は中止される（ブロック７

３６）。それ以外の場合、エンジン２４４は監視中の（ｌｉｓｔｅｎｉｎｇ）アプリケーションへの接続を示唆し（ブロック７３７）、設定を取得する（ブロック７３８）。この２ステップのどちらかが成功しなかった場合、接続は中止される（判定ブロック７３９、ブロック７４０）。成功した場合、処理はスリープ状態に戻って作業を待つ（ブロック７４１）。

【０１７６】

（異常終了の例）

図１５Ａおよび１５Ｂは、接続「中止」要求を受けて、インターネット・モビリティ・プロトコル・エンジン２４４が実行するステップを例示する、フローチャートを形成している。キューを介してディスパッチされた、上記のような要求を他の処理（ブロック９９９）から受信すると（ブロック１０００）、エンジン２４４は、接続が該要求と関係しているかどうかを判断する（判定ブロック１００１）。関係していれば（判定ブロック１００１で「イエス」の場合）、エンジン２４４は元の状態をセーブして（ブロック１００２）、「中止」状態を宣言する（ブロック１００２ａ）。次に、エンジン２４４は、接続がＲＰＣエンジンに示唆されていたか否かを判定する（判定ブロック１００３）。されていれば、エンジン２４４は切断イベントを示唆し（ブロック１００４）、「ポスト・モーテム」状態を宣言して（ブロック１００３ａ）、該接続に割り当てられていたリソースを解放し（ブロック１００５）、元の状態が保留中の状態より大きいか否かを判定する（判定ブロック１００６）。元の状態が保留中の状態より大きくなければ、（判定ブロック１００６で「ノー」の場合）、処理は、呼び出しルーチンに戻るべく、ブロック１０１２に移る（ブロック１００７）。元の状態が保留中の状態より大きい場合には、エンジン２４４は、上記要求が受信フレームと関係しているかどうかを判定する（判定ブロック１００８）。中止要求が受信フレームと関係していて、受信フレームが中止フレームである場合（判定ブロック１００９）、受信フレームは呼び出しルーチンに戻る（ブロック１０１２）前に破棄される（ブロック１０１１）。

【０１７７】

（ローミング制御の例）

再度図１を参照すると、モバイルネットワーク１０８は、それぞれ別のネットワーク相互接続（１０７ａ～１０７ｋ、無線トランシーバ１０６ａ～１０６ｋにそれぞれ対応）を提供する、複数のセグメントを含んでいてもよい。本発明の別の側面において、モビリティ管理サーバ１０２を含むネットワーク１０８は、モバイル端末システム１０４があるネットワーク相互接続から別のネットワーク相互接続へと移る「ローミング」状況を、余裕を持って（ｇｒａｃｅｆｕｌｌｙ）扱うことができる。一般に、ネットワーク１０８のトポロジは、管理等の理由により、いくつかのセグメント（サブネット）に分けられ、一般的には、あるセグメントにおいて、別個のネットワーク（伝送）アドレスが複数のモバイル端末システム１０４それぞれに割り当てられている。

【０１７８】

上記のようなサブネットで新規に起動したネットワーク装置は、動的ホスト構成プロトコル（ＤＨＣＰ）を使用して、自動的に設定されるのが一般的である。例えば、サブネット上のＤＨＣＰサーバは、一般的に、そのクライアントに（他のものの提供に加えて）有効なネットワークアドレスを「リース」する。ＤＨＣＰクライアントは、永続的に割り当てられた、「固定符号化された（ｈａｒｄ－ｃｏｄｅｄ）」ネットワークアドレスを有していなくてもよい。その代わり、ブート時に、ＤＨＣＰクライアントはＤＨＣＰサーバにネットワークアドレスを要求する。ＤＨＣＰサーバは、割り当てに使用可能なネットワークアドレスをプールしている。ＤＨＣＰクライアントがネットワークアドレスを要求すると、ＤＨＣＰサーバはプールしていたアドレスを、該クライアントに割り当てあるいはリースする。割り当てられたネットワークアドレスは、特定の期間（リース期間）だけ、クライアントに所有される。リース期間が終わると、ネットワークアドレスはプールに戻され、他のクライアントへの割り当てに使用できるようになる。ネットワークアドレスの自動割り当てに加え、ＤＨＣＰはネットマスク等の設定情報を、ＤＨＣＰクライアントソフト

ウェアを動作させているクライアントに提供する。標準的なＤＨＣＰプロトコルについての詳しい情報は、ＲＦＣ２１３１に記載されている。

【０１７９】

よって、ＤＨＣＰを使用するモバイル端末システム１０４が、サブネットからサブネットへとローミングすると、該システムは新規のネットワークアドレスを持つことになる。本発明の一側面において、モバイル端末システム１０４とモビリティ管理サーバ１０２とがＤＨＣＰの自動設定機能を利用し協調することで、モビリティ管理サーバによるモバイル端末システム１０４の「新しい」ネットワークアドレスの認識や、該アドレスと、モビリティ管理サーバがプロキシする、以前に確立されていた接続との結合が保証される。

【０１８０】

ある実施例では、モバイル端末システム１０４が別のサブネットにローミングしたり圏外に出たりしたか否かを判定するために利用される、他の標準的な方法と並んで、エコー要求－応答（ｅｃｈｏ　ｒｅｑｕｅｓｔ－ｒｅｓｐｏｎｓｅ）として、標準的なクライアント／サーバ型同報通信用のＤＨＣＰ　Ｄｉｓｃｏｖｅｒ／Ｏｆｆｅｒメッセージのシーケンスが使用される。標準的なＤＨＣＰプロトコルに従って、ネットワークアドレスを要求するモバイル端末システム１０４が、ＤＨＣＰ　Ｄｉｓｃｏｖｅｒメッセージの一部として、クライアント識別子とハードウェアのアドレスとを定期的に同報通信すると、これに対し、ＤＨＣＰサーバはＯｆｆｅｒ応答を同報通信する（要求しているモバイル端末システムがまだネットワークアドレスを持っていないため、該システムを特定して該応答を送るというよりも、同報通信によって該応答が伝えられるという形になる）。よって、当該サブネット上の全てのモバイル端末システム１０４は、ＤＨＣＰ　Ｏｆｆｅｒサーバからの、同サブネット上のどのモバイル端末システムに同報通信された応答であっても受信することになる。

【０１８１】

本実施例では、ＤＨＣＰ同報通信メッセージをモニターするＤＨＣＰリスナが設けられており、モバイル端末システム１０４が、あるサブネットから別のサブネットへローミングしているか、およびＤＨＣＰによって新規のネットワークアドレスが取得できるようになっているかが確認される。図１６は、ＤＨＣＰリスナのデータ構造の例を示している。例えば、モバイル端末システムのリスナのデータ構造９０２は、以下の要素を有していてもよい。

【０１８２】

・サーバデータ構造の連結リスト、
・整数トランザクションＩＤ番号（ｘｉｄ）、
・カウンタ（「ｐｉｎｇ」）、および
・タイムアウト値

サーバデータ構造９０４は、それぞれ別のＤＨＣＰサーバを定義するデータブロックの連結リストを有していてもよい。該データブロックそれぞれは、以下の要素を有していてもよい。

【０１８３】

・次のサーバへのポインタ、
・サーバＩＤ（ＤＨＣＰサーバのネットワークアドレス）、
・最近該ＤＨＣＰサーバと結合したＢＯＯＴＰリレーエージェントのアドレス（ｇｉａｄｄｒ）、
・「ｐｉｎｇ」値（ｓｏｃｋｅｔ－＞ｐｉｎｇ）、および
・フラグ

これらのデータ構造は、ネットワーク１０８上のＤＨＣＰ同報通信トラフィックに基づいて、持続的にアップデートされる。以下に例示の諸機能は、該データ構造の維持に使用される。

【０１８４】

・ｒｏａｍＣｒｅａｔｅ（）　（変数を初期化）

・ｒｏａｍＤｅｉｎｉｔｉａｌｉｚｅ（）　（全てのリスナを消去）
・ｒｏａｍＳｒａｒｔＩｎｄｉｃａｔｉｏｎｓ（）　（モバイル端末システムがローミングするかインターフェイスを変更した際に、登録主体ローミング示唆（ｒｅｇｉｓｔｒａｎｔ　ｒｏａｍｉｎｇ　ｉｎｄｉｃａｔｉｏｎｓ）のために、供給されたコールバックルーチンを呼び出す）
・ｒｏａｍＳｔｏｐＩｎｄｉｃａｔｉｏｎｓ（）　（適当なコールバックをリストから除去して登録ローミング示唆を停止）
・インターフェイス変更（インターフェイスがネットワークアドレスを変更したことを示す、ＯＳからのコールバック通知）
・リスナ信号（ローミング、圏外、圏内復帰のいずれかの状態を示す、リスナからのインターフェイスごとのコールバック）
これに加え、リフレッシュ処理が、インターフェイス変更後にリスナをアップデートするために利用されてもよい。

【０１８５】
本好ましい実施形態では、全てのモバイル端末システム１０４が、ＤＨＣＰ　Ｄｉｓｃｏｖｅｒ要求によって同じクライアント識別子とハードウェアアドレスとを伝送する。これにより、リスナのデータ構造とこれに関連する処理とが、モバイル端末システムからのＤｉｓｃｏｖｅｒ要求と、他のネットワーク装置からのＤｉｓｃｏｖｅｒ要求とを、区別することができるようになる。ＤＨＣＰサーバも同様に応答を同報通信するので、どのモバイル端末システム１０４および／またはモビリティ管理サーバ１０２も、ＤＨＣＰサーバからどのモバイル端末システムに出されたＯｆｆｅｒ応答でも受け取ることができる。複数のＤＨＣＰサーバが単一のＤＨＣＰ　Ｄｉｓｃｏｖｅｒ要求に応答できるため、図１６のリスナのデータ構造は、サーバからの応答それぞれを、メインハンドルに連結リスト経由で結びつけられた、個別のデータブロックに記憶する。

【０１８６】
所定のクライアントハードウェアアドレスおよびクライアント識別子を有するＤｉｓｃｏｖｅｒ要求の受信に際し、本好ましい実施形態では、該要求がモバイル端末システム１０４から送られてきたものとして認識される。該メッセージが０に設定されたＢＯＯＴＰリレーアドレスをさらに有している場合、該メッセージはリスナと同じサブネットからのものであることが示唆されている。リスナは、最近モバイル端末システム１０４から送られたＤｉｓｃｏｖｅｒメッセージのトランザクションＩＤ（ｘｉｄ）と対応したトランザクションＩＤ（ｘｉｄ）が含まれていない限り、ＤＨＣＰ　Ｏｆｆｅｒ応答を無視してもよい。該リスナは、新規のＢＯＯＴＰリレーエージェントＩＤおよび／または提供されたサブネットマスクでマスクされた、提供されたネットワークアドレスを持つ、既知のサーバから応答があれば、モバイル端末システム１０４がローミングしたと判定する。リスナは、旧サーバから肯定応答を受け取って初めて、新サーバを図１６のデータ構造に加える。リスナが新サーバからの応答は受け取ったが旧サーバからは受け取っていない場合、ローミング状態が示唆される（これについては設定によって変更可能である）。リスナが新旧両サーバのどちらからも応答を受け取っていない場合、リスナは圏外にあると判定される（この判定は、アプリケーションなど上層に警告を出して、停止や、バッファ・オーバーフロー回避のためのデータ送信量の減少に利用できる）。

【０１８７】
リスナがどのサーバからも応答を受信しない場合、参照点が無いため、ローミングが行われているかどうかが判定できない。この状況は、タイムアウト後にエラー警告して、呼び出し側に処理を再試行させることによって解消される。本好ましい実施形態では、新規のＢＯＯＴＰリレーエージェントＩＤ（または提供されたサブネットマスクでマスクされた、提供されたネットワークアドレス）を持つ、既知のサーバから応答があれば、モバイル端末システム１０４がローミングしたと判定する。リスナのデータ構造に、新サーバからの応答はあったが旧サーバからのものは無い場合、ローミングが行われた可能性はあるが、旧サーバからの応答がその後あるかもしれないので、待機して通知を遅らせる。新旧ど

ちらのサーバからも応答が無い場合、モバイル端末システム１０４は圏外にある可能性があるので、モビリティ管理サーバ１０２は該システムが圏内に戻るのを待つ。

【０１８８】

図１７は、本好ましい実施形態のリスナ処理のステップを例示したフローチャートである。同図によると、ＤＨＣＰリスナ処理は、適切なメモリをハンドルに割り当て、ＮＡＬソケットをＤＨＣＰクライアントおよびサーバＵＤＰポートに解放し、その両者に対して受信コールバックを設定することによってなされる。次にタイマーが設定され（ブロック８０２）、上記処理は「待機」状態に入ってローミング関連のイベントを待つ（ブロック８０４）。イベントは、以下の３種の外部入力によって引き起こされる。

【０１８９】

・ＤＨＣＰサーバパケットを受信する

・他のモバイル端末システムからのＤＨＣＰクライアントパケットを受信する　・タイマーの期限が切れる

ＤＨＣＰサーバパケットを受信した場合、該パケットは、そのクライアント識別子が所定のクライアントＩＤと一致するか否かを判定するために調べられる（判定ブロック８０６）。一致しなければ、該パケットは破棄される。しかし、該パケットが所定のクライアントＩＤを含んでいる場合、該パケットがＤＨＣＰ　Ｏｆｆｅｒパケットであるか否かが判定される（判定ブロック８０８）。該Ｏｆｆｅｒパケットは、最近送られたＤＨＣＰ　Ｄｉｓｃｏｖｅｒシーケンスに対応したトランザクションＩＤを含んでいない限り、拒絶される。

【０１９０】

パケットトランザクションＩＤが対応していれば（ブロック８１０）、ＤＨＣＰ　Ｏｆｆｅｒパケットを送信したサーバが既知であるか否か（つまり、サーバＩＤが図１６のリスナのデータ構造に含まれているか否か）が判定される（ブロック８１２）。サーバＩＤがリストに無い場合（判定ブロック８１２で「ノー」の場合）、サーバＩＤがリストに加えられ、「新規」とマークされる（あるいは、リストの最初のサーバであれば、「最初」とマークされる）（ブロック８２２）。サーバが既にリストにある場合（判定ブロック８１２で「イエス」の場合）、さらに、パケットＢＯＯＴＰリレーアドレス（「ＧＩＡＤＤＲ」）がサーバアドレス（「ＧＩＡＤＤＲ」）に対応しているか否かが判定される（判定ブロック８１４）。対応していなければ、Ｏｆｆｅｒパケットは他のサブネットからのものということになるので、「ハードローミング（ｈａｒｄ　ｒｏａｍ）」が実行されたと判定される（ブロック８１６）。呼び出し側のアプリケーションには、ローミングが行われたことが通知される。判定ブロック８１４でＢＯＯＴＰリレーアドレスが対応していると判定されると、ローミングは行われていないということになり、サーバ受信時間を記録し、リストの他のサーバ全てについて「新規」フラグをリセットし、現在のｐｉｎｇ番号をサーバに記憶するというリスナ処理が行われる（ブロック８１８、８２０）。続いて、処理は「待機」期間に戻る。

【０１９１】

イベントがクライアントパケットに受信されると、リスナ処理では、該パケットが所定のクライアントＩＤを有しているか、ＤＨＣＰ　Ｄｉｓｃｏｖｅｒパケットか、および０のパケットＢＯＯＴＰリレーアドレス（ＧＩＡＤＤＲ）を有しているかどうかが判定される（ブロック８２４、８２６、８２８）。これらのステップにより、上記受信パケットが、リスナと同じサブネット上にある、他のモバイル端末システムから送信されたＤＨＣＰ　Ｄｉｓｃｏｖｅｒメッセージであるか否かが判定される。そうであれば、リスナ処理により、上記トランザクションＩＤが、後に受信されるＤＨＣＰ　Ｏｆｆｅｒパケットの比較に用いられる、ピアのトランザクションＩＤに設定され（ブロック８３０）、「ｐｉｎｇ確認」が呼び出され（ブロック８３４）、タイマーがリセットされる（ブロック８３６）。

【０１９２】

タイマーの期限切れにより、処理は「ｐｉｎｇ確認」を呼び出す（ブロック８３８）。本

好ましい実施形態における「ｐｉｎｇ」は、ランダムな新規のｘｉｄを有するＤＨＣＰＤｉｓｃｏｖｅｒパケットである。このｐｉｎｇ確認８３８のステップが、図１７Ａに例示されている。ｐｉｎｇ確認ルーチンの目的は、「ソフトローミング（ｓｏｆｔ　ｒｏａｍ）」状態（つまり、モバイル端末システムが、一時的にサブネットとのコンタクトを失ったもののその後回復したが、まだ他のサブネットにローミングはしていない状態）が発生しているか否かの判定である。処理によって、サブネット・ローミング状態、圏外状態、あるいは「サーバ無し」状態が発生しているか否かが判定される。これらを言い換えると以下のようになる。

【０１９３】

・モバイル端末システムが、あるサブネットから別のものにローミングしたか？

・モバイル端末システムは圏外にあるか？

・ＤＨＣＰサーバは不在か？

これらの状態は、モバイル端末システムの、前「ｐｉｎｇ」応答と現「ｐｉｎｇ」応答とを比較することによって判定される（判定ブロック８４６、８５０）。例えば、現ｐｉｎｇ数から旧サーバの前ｐｉｎｇ応答を引いた数が、サブネットのサーバのｐｉｎｇよりも大きく、少なくとも一つのサーバが「新規」とマークされている場合、別のサーバへのサブネット・ローミングがあったということになる。この論理により、コーリング処理に対し、サブネット・ローミング状態、圏外状態、あるいはサーバ無し状態が示唆される（あるいはいずれも示唆されない）。

【０１９４】

図１８は、モバイル端末システム１０４のローミング制御センター（ｒｏａｍｉｎｇ　ｃｏｎｔｒｏｌ　ｃｅｎｔｅｒ）が実行するステップを例示したフローチャートである。モバイル端末システム１０４でのローミングを可能にするため、既知のアドレスのリストが０に初期化され（ブロック８５０）、ＯＳインターフェイス変更通知がイネーブルになり（ブロック８５２）、次に、ＯＳを呼び出して、ＤＨＣＰを利用する現在のアドレスのリストを所得する（ブロック８５４）。現在のリストから無くなった全ての既知のアドレスに対応するリスナが閉じられ（ブロック８５６）、一方、現在のリストにあるが未知のインターフェイスにあるリスナが開放される（ブロック８５８）。次に、登録主体に「ローミング」を示唆する（ブロック８６０）。

【０１９５】

図１７のリスナ処理から信号が送られると（ブロック８６２）、該信号が、「ローミング」、「圏外」、「圏内復帰」のいずれの状態を示しているかが判定される（判定ブロック８６４、８７０、８７４）。ローミング信号（判定ブロック８６４で「イエス」）により、対応するリスナ８６６が閉じられ、ＯＳがコールされて、ネットワークアドレスのＤＨＣＰリースが解放、更新される（ブロック８６８）。リスナ信号が「圏外」の場合（判定ブロック８７０）、該状態が登録主体に示唆される（ブロック８７２）。該信号が「圏内復帰」の場合（判定ブロック８７４）、該状態は全ての登録主体に示唆される（ブロック８７６）。無効のローミング命令を受信すると（ブロック８７８）、全てのリスナが閉じられ（ブロック８８０）、ＯＳインターフェイス変更通知が無効になる（ブロック８８２）。

【０１９６】

（インターフェイスによって補助されたローミングのリスナの例）

さらに、インターフェイスベースのリスナによって、同じネットワークおよび別のネットワーク媒体のネットワーク接続ポイントを横断したローミングが可能になる。該インターフェイスベースのリスナが上述のビーコン技術を要することなく動作する一方、下層の（複数の）インターフェイスが適切な信号をサポートしていない場合には、システムをビーコン状態にフォールバックさせることもできる。

【０１９７】

本実施形態において、インターフェイスベースのリスナは、ネットワーク・インターフェイス・アダプタからの（例えば低レベルのインターフェイス・ローミング・ドライバを経

由した）情報を、ネットワークスタックからの情報と統合して、モバイルノードが新規のネットワーク接続ポイントに移動したか否かを判定する。図１９Ａと１９Ｂは、モバイルノードの移動パス（ｍｉｇｒａｔｉｏｎ　ｐａｔｈ）を効率的に決定するのに利用される、リスナ・アルゴリズムを例示する。該処理では、単一のネットワーク媒体に接続された単一のネットワーク・インターフェイスが使用されているが、単独で、もしくは他のローミング・アルゴリズムと協働して、（例えば、冗長パスを使った自己回復インフラ構築のために）様々なネットワーク媒体やインターフェイスを横断するようにもできる。

【０１９８】

図１９Ａを参照すると、システム初期化時や、ネットワークアダプタドライバがロードする際に（図１９Ａ、ブロック２０００）、低レベルのインターフェイス・ローミング・ドライバは、図１８のローミング制御センターモジュールに登録を実行する（ブロック２０１０）。このような登録（例示の実施形態では、ｃｒＲｅｇｉｓｔｅｒＣａｒｄＨａｎｄｌｅｒ（）機能を通じて実現される）により、下記のエントリポイントが与えられる。

【０１９９】

・開
・閉
・状態（ｓｔａｔｕｓ）取得
・ドライバが登録主体に状態の変化を通知可能であれば、ブール演算を真とし、ローミング制御センターモジュールが、状態確認にタイマーベース（等）のポーリングを使わなければならない場合は、ブール演算を偽とする

実施例のｃｒＲｅｇｉｓｔｅｒＣａｒｄＨａｎｄｌｅｒ（）機能により、インターフェイス記述ストリング、あるいはローミング制御センターモジュールを正しいローミングドライバと予備的に組み合わせるために使用できるトークンが与えられる。また、デフォルトのローミングドライバが、示唆／問合せ（ｓｉｇｎａｌｉｎｇ／ｑｕｅｒｙｉｎｇ）媒体接続性およびネットワーク接続ポイントの変更に関するＯＳ包括メカニズム（Ｏ／Ｓ　ｇｅｎｅｒｉｃ　ｍｅｃｈａｎｉｓｍ）を使用するインターフェイスに、インストールされてもよい。

【０２００】

本実施例では、インターフェイスの状態がイネーブルになると（つまりネットワークへのアクセスが可能になると）（ブロック２０２０）、ローミング制御センターは、インターフェイス補助ローミング（ｉｎｔｅｒｆａｃｅ　ａｓｓｉｓｔｅｄ　ｒｏａｍｉｎｇ；ＩＡＲ）を、以下のステップに基づいて試みる（ただし、以下のステップは、ＯＳの設計および／または特定のアプリケーションに使われるホスティング装置によって、入れ替えられたり省略されたりすることがある）。

【０２０１】

１．包括ハンドラ（ｇｅｎｅｒｉｃ　ｈａｎｄｌｅｒ）がインストールされている場合、包括ｃｒＯｐｅｎＩｎｓｔａｎｃｅ（）ハンドラへのコールがなされる。包括ハンドラは低レベルアダプタドライバに問合せをして、該ドライバが、媒体接続性の状態およびネットワーク接続ポイントの変更に関する信号を包括的にサポートしているか否かを判定する（ブロック２０３０）。インターフェイスドライバが該機能を包括的にサポートできない場合（判定ブロック２０３０で「ノー」の場合）、エラー状態が、コール実行側に返され、信号情報の取得に他のメカニズムを使用することが示唆される。

【０２０２】

２．包括ハンドラがエラー（判定ブロック２０３０で「ノー」）を返した場合、アクティブなインターフェイスに関する検索が、現在登録されているローミングドライバにおいて実行される（ブロック２０４０）。該インターフェイスが、ｃｒＲｅｇｉｓｔｅｒＣａｒｄＨａｎｄｌｅｒ（）フェイズ中に登録されたトークンのうちの一つと一致する場合（ブロック２０５０）、ローミング制御センターは、アダプタのインスタンスへの特定のｃｒＯｐｅｎＩｎｓｔａｎｃｅ（）をコールする。この機能は、低レベルドライバを開き、状態（媒体接続性、ネットワーク接続ポイントＩＤ）を再度ポーリングし、定期ポーリング

タイマーを（可能ならば）設定することを試みるものである。低レベルドライバが、なんらかの理由で該要求をサポートしていない場合、ローミング制御センターにエラーが返されて、信号情報の取得に他のメカニズムを使用することが示唆される。

【０２０３】

３．ここまでのステップのいずれかが、要求された機能を達成できない場合、エラーがローミング制御センターに返されて、ＩＡＲ機能を使用せずに、図１７および１７Ａのビーコンリスナ（ｂｅａｃｏｎｉｎｇ　ｌｉｓｔｅｎｅｒ）、モバイルＩＰ、あるいは場合により、ローミングを扱っている、現在接続されているネットワークなど、他のローミング・アルゴリズムにフォールバックすることが示唆される（判定ブロック２０５０で「ノー」の場合、ブロック２０６０）。これ以外の場合、インターフェイス補助ローミングがイネーブルになり（ブロック２０６０）、ローミング制御センターは下記のアルゴリズムに従う。

【０２０４】

まず、インターフェイスによって補助されたリスナは、現在の媒体接続性の状態と、ネットワーク接続ポイントの識別情報とを、ローカルデータストアに記録する（ブロック２０６０）。インターフェイスによって補助されたサブシステムがローミングフィードバックの供給に成功したと仮定して、該サブシステムは状態イベントを待つ（ブロック２１００）。該イベントは、例えば以下のものを有している。

【０２０５】

・低レベルローミングドライバからのコールバック、

・ポーリング間隔（ｔｉｍｅｄ　ｐｏｌｌ　ｉｎｔｅｒｖａｌ）（ブロック２０７０、２０９０）、あるいは

・ネットワーク・レベルの活動（つまり、伝送／受信にかかわる問題）からの示唆

インターフェイスの状態が、媒体接続性に変化が生じたか、あるいはネットワーク接続ポイントが変更されたかを示すと（図１９Ｂの判定ブロック２１１０もしくは２１２０で「イエス」の場合）、ローミング制御センターの全てのクライアントに、以下のルールに基づいて、状態の変化が知らされる。

【０２０６】

１．状態が、下層のネットワーク媒体への接続から切断へと変化したことを示し（ブロック２１２０で「イエス」）、ピアへのパスが他に無い場合、リスナは、モバイル端末システムが接続を失ったと結論付け、ローミング制御センターはそのクライアントに、ＲＯＡＭ＿ＳＩＧＮＡＬ＿ＯＵＴ＿ＯＦ＿ＣＯＮＴＡＣＴ状態を示唆する（ブロック２１４０）。

【０２０７】

２．上記状態が、インターフェイスが媒体に再接続され、ネットワーク接続ポイントが変わっていないことを示し（ブロック２１２０で「ノー」の後、ブロック２１５０で「ノー」の場合）、ＲＯＡＭ＿ＳＩＧＮＡＬ＿ＯＵＴ＿ＯＦ＿ＣＯＮＴＡＣＴが既に示唆されている場合、モバイル端末システムが、特定のネットワーク接続ポイントとのコンタクトを失ったがその後取り戻したことが示唆される。この場合、ローミング制御センターは、適切なアクセスのために登録または取得された全てのネットワークアドレスを再確認し（ブロック２１７０）、ＲＯＡＭ＿ＳＩＧＮＡＬ＿ＲＯＡＭ＿ＳＡＭＥ＿ＳＵＢＮＥＴを出して（ブロック２１８０）、ローミング制御センターのクライアントに、再接続が行われたので、トランスポート・レベルでの通信の再確立に必要な措置を全て実行するように警告する。例えば、サービス中断中にいくつかのデータが失われることがあれば、クライアントは該データを回復するために措置を講じるといったことである。

【０２０８】

３．上記状態が、インターフェイスが媒体に取り付けられているが、ネットワーク接続ポイントが変更されたことを示している場合（ブロック２１５０で「イエス」の場合）、ローミング制御センターはクライアントに、ローミング状態になったことを示唆する。ネットワーク接続ポイント間のハンドオフをさらに効率的にサポートするために、本例のロー

ミング制御センターは、ローカルデータストアと並行して学習アルゴリズムを使用している。該データストアは、通常動的にポピュレートされている（学習している）が、パフォーマンス向上のため、該データストアに静的な情報（既に学習された情報）が準備されていてもよい。データストア自体は、ネットワーク接続ポイント識別子のリストを、ネットワークや媒体へのアクセスのアドレスやネットワークマスクなどの情報とともに維持している。この「ネットワーク・トポロジ・マップ」は、ローミング制御センターがクライアントに対して正しい信号の生成を決定できるように補助するものである。

【０２０９】

正しい信号の決定は、本実施例では以下のようにしてなされる。

【０２１０】

ａ）データストアのネットワーク・トポロジ・マップを検索して、インターフェイスが特定のネットワーク接続ポイントを訪れたか否かが判定される（ブロック２１９０）。対応が見つかれば（ブロック２２００で「イエス」）、該ネットワーク接続ポイントが、以前にインターフェイスが結合していたのと同じネットワーク・セグメントにあるか否かがさらにチェックされる。ネットワーク・セグメントが同じであれば、ローミング制御センターは、ＲＯＡＭ＿ＳＩＧＮＡＬ＿ＲＯＡＭ＿ＳＡＭＥ＿ＳＵＢＮＥＴを生成する。これにより、ローミング制御センターのクライアントに、ハンドオフが実行され、ハンドオフ中にデータのいくつかが失われた可能性があるため、直ちにトランスポート・レベルでの通信の再確立をするための可能な限りの措置を講じなければならないことが示唆される。

【０２１１】

ｂ）検索中に対応は発見されたが、新しいネットワーク接続ポイントは前と同じネットワーク・セグメントにない場合、リスナは、モバイル端末システムが別のサブネットワークにローミングしたと結論付ける。この場合、ローミング制御センターは、

・新規のネットワーク・セグメントで使用可能なアドレスを取得する（ブロック２２２０）。この動作は、現在のアドレスを新規のセグメントで有効なように登録すること、ローカルサーバからアドレスを（再）取得すること、あるいは以前に割り当てられたアドレスがまだ有効かどうか判定するヒューリスティックな手法を用いてもよい。最後のケースでは、ローミング制御センターにより、インターフェイスが所与のネットワーク接続ポイント間をローミングしていて、パフォーマンス維持の観点から、直ちにネットワークアドレスを破棄したり、その登録を抹消したりできないような状態にあるかどうかが判断される。この例では、（例えばＤＨＣＰを通じて）ネットワークでアドレスを取得することは、（モバイルＩＰの外部エージェントを通じて）ローカルネットワークでアドレスを登録することとは相違している。ローミングエンティティは、アドレスを（再）取得（例えばＤＨＣＰサーバからのリースを確立／アップデートして）するか、あるいは現アドレスを外部エージェント（モバイルＩＰ）に登録する。

【０２１２】

・別のサブネットへのローミングを示唆する、ＲＯＡＭ＿ＳＩＧＮＡＬ＿ローミング信号を、該ローミング制御センターのクライアント向けに生成する（ブロック２２３０）。

【０２１３】

ｃ）検索の結果対応が見つからなければ（ブロック２２００で「ノー」）、ネットワーク接続ポイントの識別子、媒体アクセスアドレス、ネットワークマスクや他の補助的な情報によってポピュレートされた、新規の記録を生成する（ブロック２２１０）。次に、ローミング制御センターはブロック２２２０および２２３０を、ネットワークアドレスを取得、登録し、「ローミング」信号を生成するために実行する。

【０２１４】

上記のインターフェイスによって補助されたローミング技術により、下層のインターフェイス情報へのアクセスができるようになり、自動的かつ効率的に別の有効なネットワークパスを選択することを可能にする、（ユーザおよび／またはシステムによって定義された）付加的なポリシーパラメータが採用できるようになる。一つ以上のネットワークが同時に使用可能な場合、サブシステムは、最も負担がかからないパス（ワイドエリア・ネット

ワークかローカル・エリア・ネットワークかという選択）を選ぶことができる。これは、帯域幅、（１バイトあたりの）コスト、および／またはサービス品質といった様々なメトリクスによって決定可能である。この「最低負担ルーティング」技術により、ネットワーク接続の品質、効率、フレーム損失の減少などの点で効果を得ることができる。例えば、他の利用可能なヒューリスティクス（媒体接続性、信号強度、再伝送率等）による、「メークビフォアブレーク（ｍａｋｅ　ｂｅｆｏｒｅ　ｂｒａｋｅ）」ハンドオフ構造が提供可能で、よってローミングノードからの／に向かう継続的なパケットフローの損失を最小化できる。下記のポリシー管理についての記述を参照のこと。

【０２１５】

図２０は、インターフェイス補助ローミングトポロジのノードのデータ構造を例示している。図２０のこのデータ構造は連結リストとして実施されているが、前後のフィールドを省略した配列として表現されることもある。無線ネットワーク・インフラにおいて、「ＮＰＯＡ」は、例えば、アクセスポイントのＭＡＣアドレスあるいはモバイルノードが結合している基地局でもよい。他のネットワークにおいては、「ＮＰＯＡ」は、介入的なネットワーク相互接続（ゲートウェイ、ＩＷＦ等）の固有の識別子であってもよい。データ構造には静的な情報が予め与えられていてもよいし、動的に情報が学習されてもよい。また、ノードそれぞれと、他の情報（例えばＭＴＵのサイズ、待ち時間、コスト、利用可能かどうかなど）とが結合していてもよい。

【０２１６】

（特定の競合状態を扱う他の実施例）

さらに行われた実験から、ネットワーク・アダプタの中には、ネットワーク・セグメントに完全登録される前に、媒体と（再）接続されていると誤って信号を出すものがあることが明らかとなった。例えば、ローミングの間、ネットワーク識別子を保持する記憶領域はまだ更新されず、従って、システムがこれらのネットワーク・アダプタが同じサブネットにロームバックしたと誤解することが考えられる。最終的には、デバイスが登録を終了すると、記憶領域は新しいネットワーク識別子と共に更新され、さらに別のローミング信号が生成される。両方の情報が一緒にゲートされ、インターフェイスがネットワークでの登録を終了した時に１度だけ信号が送られると、このシナリオ通りに進む。しかし、ポーリングの際、「ネットワークと接続中」を示す信号が以前に生成された場合に、ネットワークＩＤがいつ有効になるかを判定するのは困難である。

【０２１７】

基本的に、ローミング中のノードはネットワークと媒体アクセスレベルで通信できるため、事実上媒体と接続状態にあるが、登録プロセスが完了していないので、事実上、リンクを通じていずれのアプリケーションデータを送ることはまだできない。従って、この状態を補償するのが望ましい。このような補償を行う方法の一つとして、一般にはエコー要求／応答パケットとして知られているリンク確認フレームを送ることによってピア接続を判定する方法がある。これらのエコーやｐｉｎｇフレームは１つのピア（ローミング中のノードである可能性が高い）によって生成され、双方向のピア・トゥ・ピア接続が達成可能か判定する。要求を出しているピアがその要求に対する応答フレームを受け取る場合、二重通信が実現し終了する。この時点で、次に切断状態になるまでＮＰＯＡ情報が有効であると見なされる。また、該当インターフェイス上のピアからのフレーム受領等、他の情報によって、ローミング中のノードが登録プロセスが終了し、双方向通信が達成可能であることが想定可能となる。

【０２１８】

ネットワーク・インターフェイスと内在するプロトコルスタック状況間における別の競合状態は、問題を生じることがある。デバイスは、新しいネットワーク・セグメントに移動し、以下のインターフェイスから正確に信号が送られるようにすることは可能であるが、トランスポート・スタック自体はフローするアプリケーションデータのルーティング・テーブルに必要な調整を行わないことがある。この状態を補償するために、内在するトランスポートのルーティング・テーブルに変更が生じる度に、付加的信号であるＲＯＡＭ＿Ｓ

ＩＧＮＡＬ＿ＲＯＵＴＥ＿ＣＨＡＮＧＥが追加されて生成される。この信号が示されると、ローミング中のサブシステムクライアントはピア・システムとの接続が達成可能かどうか判定するために必要なアクションは何でも行う。これによりローミング中のクライアントは、ルーティング修正がピアへの通信経路に影響を及ぼしたかどうか判定するために、内在するトランスポートのルーティング・テーブルを介して列挙する必要がある。また、上記記載のアルゴリズム等のように、よりイントルーシブな他のアルゴリズムが、ピア間に双方向通信経路が存在することを確認するために行われてもよい。

【０２１９】

（非接続ネットワークを通じたローミング例）

本発明の非限定的な好ましい実施形態の他の一側面として、いわゆる「非接続ネットワーキング」（ｄｉｓｊｏｉｎｔ　ｎｅｔｗｏｒｋｉｎｇ）モードでＭＭＳ（モビリティ管理サーバ）にアクセスするためのアルゴリズム及び構成がある。新しいアルゴリズムによって、あるネットワークからは別のネットワークにおけるネットワークアドレスが分からないような非接続ネットワーク・トポロジにおいても、ＭＭＳとの通信を確立する／持続するのに使われる、代替のネットワークアドレスを動的／静的に見つけ出すことができるようになる。

【０２２０】

一般に、アルゴリズムによって、通信中にＭＥＳ（モバイル端末システム）に送るための、ＭＭＳが利用可能な代替アドレスのリストが可能となる。このように、ＭＭＳはＭＥＳに、一つ以上のＭＭＳネットワークアドレスもしくは他のネットワークに対応した他のＭＭＳのアイデンティティを、単一のネットワークによる通信によって送る。一例として、該リストは、回路構築の際に送付可能である。また、該リストは途中で変更可能である。この場合、該リストは接続が確立されている間のいかなる時にも更新が可能である。

【０２２１】

ＭＥＳが別のネットワークへと移動するとき、ＭＥＳは新規のネットワークの接続ポイントからＭＭＳとコンタクトをとるために、ＭＭＳの「エイリアス」（ａｌｉａｓ）アドレス／アイデンティティのリストを用いる。これにより、移動前のネットワークと移動後のネットワークとがアドレス、又はその他の情報を共有していなくても、ＭＥＳは新規のネットワーク接続を通じてＭＭＳとのコンタクトを再確立することができる。

【０２２２】

この新しい技術を簡略化したフローチャートを図２１に示す。ＭＭＳ１０２は、２つの異なる非接続ネットワーク、すなわちネットワーク・セグメントＮ１及びＮ２と接続しているものとする。ＭＥＳ１０４は、最初にＭＭＳ１０２とネットワークＮ１を介して接続されるものとする。ネットワークＮ１を介してＭＥＳ１０４とＭＭＳ１０２との間で接続が確立されるとすぐに、ＭＭＳ１０２は、ＭＥＳ１０４に対して、ＭＭＳが一以上の他のネットワーク（例えば、ネットワークＮ２）から求められるネットワークアドレスのリストＬ又は他の識別子を送ることができる。ＭＥＳ１０４は、該リストＬを受け取り、記憶する。その後、ＭＥＳ１０４は別のネットワーク（Ｎ２）に移動する時、ＭＥＳ１０４はこの記憶されたリストＬにアクセスし、それを用いて新規のネットワーク（Ｎ２）を通じたＭＭＳ１０２との通信を効果的に再確立することが可能となる。

【０２２３】

この新しいアルゴリズムには、非接続ネットワークを通じて、ＭＭＳ１０２と通信するための代替ネットワークアドレス又は他の識別子をより効率的に取得することに加え、少なくともいくつかの用途がある。その用途の一例としてネットワークオペレーションがある。例えば、図２１に示すアルゴリズムを用いて、多くのネットワーク（いくつか、又は全てが無線ネットワーク）からの安全なファイアウォール／ゲートウェイ、及びコーポレートバックボーンとしてＭＭＳ１０２が使用される安全なネットワークをセットアップでき、モバイルノード１０４が安全に、接続を切断されることなく別々のネットワーク全てに移動することが可能となる。例えば、ＭＭＳ１０２がハブとして１つの太いパイプでコーポレートネットワーク、及び論理的に分離した多くのネットワークを接続する多数の小さ

なスポークと接続しているとする。ネットワークは論理的に分離しているため、ＭＭＳ１０２（この例ではルータとして動作可能）を介した場合を除いて、１つのネットワーク・セグメント上のトラフィックが別のネットワーク・セグメントに達することができない。

【０２２４】

普通、ネットワーク・セグメントからネットワーク・セグメントへ移動するノードにとって、ＭＭＳ１０２との接続に使用される「メインパブリックアドレス又は初期アドレス」への戻り方を示す、各ネットワーク・セグメントに設けられた情報／経路（つまり、デフォルトルート等）をルーティングが必要となる。

接続が確立されるとすぐに、そのアドレスが接続の存続のために用いられる。ＭＥＳ１０４からフレームが送られる時、クライアント及び及び中間ノード（ルータ）におけるＩＰネットワーク（層３）のインフラストラクチャは、フレームの目的地アドレスを見て、最終目的地（ＭＭＳ１０２）にパケットを正確に送る。このことは、一般にＩＰ送信（ＩＰ　ｆｏｒｗａｒｄｉｎｇ）、あるいはＩＰルーティングと呼ばれるもので行われる。この機能がオンになると、あるネットワーク・セグメントのフレーム（ブロードキャスト等）が別のネットワーク・セグメントへ流れる。ＩＰ送信をしないと、あるセグメントに送られたフレームは他のセグメントへ送られることはないので、通信パイプが切断され、あるいは非接続ネットワークができてしまう。

【０２２５】

図２１に示す代替アドレスリストは、ルーティング情報のうちの幾つかをＭＥＳ１０４に送出する、又は配布する効果を有する。従って、各セグメントは、ＭＭＳ１０２とつながっている他のセグメントの情報はない状態で離散された状態となる。ＭＥＳ１０４はＭＭＳ１０２によって認証可能であるため、ＭＭＳは認証されたＭＥＳユニット１０４にのみリストＬを送る。ＭＥＳ１０４が別のネットワーク・セグメントへ移動する時、ＭＥＳ１０４は自動的に正しいアドレスを選択し、それを用いて途中ＭＭＳとの通信を開始／継続することができる。従って、非接続ネットワークの問題は解決され、ルーティング・インフラストラクチャの変更は必要なくなる。これにより、有効なユーザに対してのみネットワークへのアクセスを有効にすることによって、より安全なコンピュータ環境を提供することが可能となる。

【０２２６】

例えば、このようにユーザレベルのセキュリティ／暗号化と組み合わせてＭＭＳ１０２を用いることで、コーポレートバックボーンからのトラフィック、及びコーポレートバックボーンへのトラフィックを、上記のローミング技術を用いてそのセグメント上のノードを目的地としたフレームのみに限定することが可能である。フレームは選択的に暗号化でき、スポーク・ネットワーク・インフラストラクチャが有効にしたデバイスによって行われる可能性のある盗聴を阻止することができる。

【０２２７】

例を図２２に示す。図２２において、ＭＭＳ１０２は通信網やルート情報を共有しない４つの別々に分かれたネットワーク（Ｉａ、Ｉｂ、Ｉｃ、Ｉｄ）と接続されている。どの点から見ても各ネットワークＩは島となっている。コーポレートバックボーン上の有線通信を用いてネットワークの一つ（例えば、Ｉｃ）とドッキングしているＭＥＳ１０４を想定する。例えば、ＭＥＳ１０４は１９２．１６８．ｘ．ｘネットワーク上のアドレスを取得してＭＭＳ１０２と通信するものとする。

【０２２８】

ある理由によりＭＥＳが１０．１．ｘ．ｘ（Ｉａ）ネットワークに渡る、又は移動する必要があるものとする。１０．１．ｘ．ｘ（Ｉａ）ネットワークは１９２．１６８．ｘ．ｘ（Ｉｂ）ネットワークのことを知らない（つまり、（Ｉｂ）ネットワークへのルートがない）ため、ＭＥＳ１０４がその領域へ移動する時に、たとえＭＭＳが通信パイプに接続されていても通信パイプは切断される。また、モバイルノード１０４が図示される他の１０．ｘネットワークのいずれかと接続する場合においても同じことが起こる。

【０２２９】

図２１に示すアルゴリズムを用いて、接続開始時間の（あるいは他の方法で）ＭＭＳ１０２は各種非接続ネットワークＩａ、Ｉｂ、Ｉｃ、Ｉｄに関するそのインターフェイスアドレスをＭＥＳ１０４と共有し、ＭＥＳはこれらを記録する。一旦記録されると、ＭＥＳ１０４は上記ネットワークのいずれか一つに移動して新しいネットワーク・セグメントに移動したことを検出する場合、ＭＥＳは適切なネットワークアドレスを選択し、そのネットワーク・セグメントでＭＭＳと通信できる。２以上のアドレスが使用される場合、ＭＥＳ１０４は適切なアドレスを選択してスピード、コスト、有効性、ホップ等、多くのメトリクスに基づき使用する。図２１と同様のリストを受け取らなかったＭＥＳ１０４は、その「ホーム」ネットワーク以外のネットワークを通じてＭＭＳと接続することができないので、種々のネットワーク間の移動を事実上阻まれることになる。

【０２３０】

他に、図２１の技術は分散型ネットワーク・インターフェイスに用いられる。今日のネットワークにおいて、ネットワーク・アドレス・トランスレータ（Ｎｅｔｗｏｒｋ　Ａｄｄｒｅｓｓ　Ｔｒａｎｓｌａｔｏｒｓ；ＮＡＴｓ）として知られるものが開発されている。この従来の技術を利用すれば、一つの公開ネットワークアドレスを用いるだけで多くのネットワークデバイスはインターネット上の情報へのアクセスが可能となる。この技術は、単一もしくはわずかなデバイスを介してインターネットに向けられた情報やクエリを全て送ることによって上記の機能を可能とする。該デバイスはネットワーク層で要求を記録し、その後パケットのアドレスとポート情報をデバイスのアドレス／ポートのタプルに再度対応付けし、それをその目的地に送信する。インターネット、あるいは他の当該ネットワークからフレームを受け取ると同時に、デバイスはリバースルックし、そのアドレス／ポートのタプル情報を開始デバイスのものと取り替えて、フレームを送り返す。これらの対応付けはＮＡＴにおいても静的に定義される。

【０２３１】

誰かがＬＡＮ／ＷＬＡＮの内部でＭＭＳ１０２を使用するために、それをＮＡＴに置くことを想定する。現在、ＭＭＳ１０２がＮＡＴでない限り、あるいは異なるプロキシを用いてＭＭＳと通信を全て行うことによって、誰かがイントラネットの境界範囲外に移動する時、ＭＭＳと通信するアドレスがもはやアクセス可能ではないため、ＭＭＳとはアクセス可能ではない。図２１のアルゴリズムを用いれば、ＭＭＳと直接接続されていない別のインターフェイスアドレスを静的／動的に明確にすることができる。従って、上記のアルゴリズムを使用すれば、ＭＥＳ１０４は自動的に適切な非接続アドレスを選択し、イントラネットの領域外のネットワークと接続する時にそのアドレスを使用することができる。

【０２３２】

この概要を図２３に示す。ノードがインターフェイス“ｄ”からインターフェイス“ｇ”へ渡るとする。ＭＭＳ１０２にローカルインターフェイスを供給するだけではアクセスはできない。ＭＥＳ１０４は分散されたインターフェイスの優先順位を知る必要がある。それからＭＥＳ１０４は必要なアドレスを選択してインターフェイス“ｇ”上で使用する。その後、ＮＡＴ２０００は、各パケットに関するネットワークアドレス／ポート情報を、内部インターフェイス“ｃ”アドレスへ適切に変換する。ＭＭＳ１０２からＭＥＳ１０４に送られるフレームには逆の処理が行われる。

【０２３３】

（ポリシー管理及びロケーションベースのサービス例）

本発明の他の非限定的な実施形態として、多くのメトリクスに基づき付加的なセキュリティ、コスト削減、サービスを提供することが独自にできる手法を提供する。上記のＭＭＳはＭＥＳが確立する各アプリケーションのセッションと深く関わっているため、どちらの側（すなわち、ＭＭＳおよび／またはＭＥＳ）もポリシーベースのルールを適用してＭＥＳとその最終的なピアとの間の通信を適合して制御することができる。さらに、ＭＭＳおよび／またはＭＥＳはデバイスのロケール、又は近接性（ｐｒｏｘｉｍｉｔｙ）、並びにネットワークの接続に基づきアプリケーション要求を調整又は修正することができる。例えば、ＭＭＳおよび／またはＭＥＳは学習され静的に定義されて適用されたルールエンジ

ン、あるいは確立する各アプリケーションセッション、又は試みる要求に対するポリシーに基づく他のルールを含むことができる。ＭＭＳはさらにＭＥＳにこのようなルールの幾つか、全くなし、あるいは一部、および／または処理を配分して、メータリング（ｍｅｔｅｒｉｎｇ）、又は傍受者によるモバイルデバイスへの侵入（ｒｏｇｕｅ　ａｔｔａｃｋｓ）に対するセキュリティを提供する。分散型トポロジで利用可能な特定の他のポリシー管理技術とは違い、ＭＭＳが中心となり、ルールやポリシー決定を管理し、それらを通信中いかなる時にも遠隔地のデバイスに分散させる。

【０２３４】

ルール自体は、ユーザ、ユーザ・グループ、デバイス、デバイスグループ、プロセスアプリケーションアイデンティティ、および／またはネットワークの接続ポイントに基づき構成可能である。一旦定義されると（学習されると）、ルールは組み合わされて、例えば、以下のものを含む種々の異なる事象、活動、および／またはサービスを管理し、制御する。

【０２３５】

・遠隔地のデバイスへの侵入アクセスの拒否、許可、又は調整
・アイデンティティに基づく特定のネットワーク・リソースへのアクセスの拒否、許可、又は調整
・利用可能、又は許可された帯域幅へのアクセスの拒否、許可、又は調整
・他のネットワーク・リソースへのアクセスの拒否、許可、又は調整
・内容又は情報の修正、調整、又は変更

このような決定は例えば、以下の要素を含む種々の異なる要素に基づいて行われる。

【０２３６】

・モバイルデバイスについての近接性、場所、高度、および／または他の特性
・日時
・アプリケーション又はプロセスアイデンティティ、アドレス等
・アプリケーション動作（例えば、帯域幅条件）
・現在のネットワーク状態
・他の静的又は動的要素

さらに分散型アーキテクチャを利用することで、ＭＭＳは同じ決定セットを適用又は共有することができる。ポリシー管理処理および／または意思決定をＭＭＳに行わせるのは、例えばモバイル装置がエンジンを実行するのに限られた処理能力を持ったり、帯域幅の限度が適用されたり、あるいはセキュリティが目的の場合に望ましい。

【０２３７】

サンプルＭＥＳの制御に使用される可能性のあるメトリクス（ルール）の幾つかを示すテーブルの例を図２４に示す。このテーブルは静的又は動的のどちらかに内容を占めてもよいし、場合によっては通信前、通信中、又は通信後に更新されてもよい。例えば、ユーザはテーブルの項目を定義するルール・エディタ（例えば、ウィザード）や他のメカニズムを使用することができる。他の構成例において、メトリクスが学習に基づくシステムによって自動的に定義され、あるいは状態の変更に基づき動的に変更される。また、ルールにはそれぞれ優先順位が割り当てられ、テーブルの場所で暗示、または割り当てによって明確に指定される。この優先順位によってエンジンが期待動作を正確に決定することができる。付加的なユーザインターフェイスの機能によって、システム管理者やデバイスのユーザはルールエンジンに問い合わせて所定のルールセットの機能を試みることが可能となる。

【０２３８】

ポリシー管理決定の基になる、以下のメトリクスをはじめとする多数のメトリクス例を示すテーブル例を図２４に示す。

【０２３９】

・ＭＥＳの通信機能（送信のみ、受信のみ、送受信）
・ＭＥＳの要求がプロキシされるか否か

・ＭＥＳのソースポート
・ＭＥＳのソースアドレス
・ＭＥＳの目的地ポート
・ＭＥＳの目的地アドレス
・ＭＥＳのプロトコル
・利用可能な帯域幅量
・プロセス名、アイデンティティ又は他の特性
・ネットワーク名、アイデンティティ又は他の特性
・ロケーション（例えば、ＧＰＳ座標又は他のロケーション情報）
・ネットワークの接続ポイント
・ユーザの識別子、アイデンティティ又は他の特性
・他のメトリクス

上記テーブル例は完全なリストではなく、本発明は上記テーブル例のメトリクス項目の範囲に限定されるものではない。ネットワークアクセス及び権利付与に関してモバイルノードの所望の動作を示すために、該項目はこの例のように具体的であり、あるいは包括的なメカニズム（例えば、ワイルドカード）であることが可能である。

【０２４０】

図２４のテーブル例には、メトリクスに基づくポリシー管理決定の結果を示す「否定要求」項目がさらに含まれる。一例として、図２４のテーブルに示す特定の項目例は、利用可能な帯域幅が一秒間で１００，０００バイト以下に低減される場合、目的地ポート２０、２１との接続は否定され、減速されることを示す。さらに、示した特定の例において、ルール（列）３、４によってネットワークトラフィックのみがＭＭＳへ、又はＭＭＳから流れることが可能となる（プロキシが行われない他の全てのネットワークトラフィックは暗に破棄される。）。

【０２４１】

一例において、各ＲＰＣ要求又はフレームが処理される前に、ルールエンジンはオペレーションの状況を決定するよう求められる。このプロセスの結果に基づき、その要求が許可、否定、あるいは遅延される。ポリシー管理決定をするためにＭＭＳおよび／またはＭＥＳによって行われる処理のフローチャート例を図２５に示す。

【０２４２】

先に概略を説明したローミング技術と、利用可能な他のロケーション又はナビゲーション情報とを組み合わせることによって、ＭＭＳはモバイル端末システムがいつ１箇所の接続ポイントから別の接続ポイントへ移動したか検出する。ネットワーク・トポロジの環境の変化を検出するために、モバイル端末システムの機能と関連してこの情報を組み合わせることによって、ロケールによって本発明はロケーションベースの傍受及びサービスの付加的なレベルを提供することができる。

【０２４３】

この情報の可能性を十分実現させるために、インターネット・モビリティ・プロトコルとＰＲＣエンジンの両方の改良点について概要を説明する。新しいＲＰＣのプロトコル及び構成の改良点が追加され、この機能が可能となる。これらを以下に挙げる。

【０２４４】

（ロケーション変更ＲＰＣ（Ｌｏｃａｔｉｏｎ　Ｃｈａｎｇｅ　ＲＰＣ）の例）

モバイル端末システムが、インターフェイス支援型ローミング又はグローバル・ポジショニング・システムから変更を検出するといった他の方法を用いて新しい接続ポイントへ移動したことを判断した時、モバイル端末システムは、フォーマットされた「ロケーション変更ＲＰＣ要求（Ｌｏｃａｔｉｏｎ　Ｃｈａｎｇｅ　ＲＰＣ　Ｒｅｑｕｅｓｔ）」メッセージをそのピア、この場合モビリティ管理サーバに送る。「ロケーション変更ＲＰＣ」は、一以上の接続ポイント識別情報をタイプ、長さ、値の形式にフォーマットする。タイプは識別情報の種類を示し、対応するタイプはＡＳＣＩＩの４８ビットＩＥＥＥ　ＭＡＣアドレス、ＩＰＶ４　アドレス、ＩＰＶ６　アドレス、経度、緯度、高度、接続機構名を

含むが、これらに限定されない。長さは識別データのバイト長を示し、該データは実際の接続ポイント識別子を含む。「ロケーション変更ＲＰＣ要求」を受け取ると、モビリティ管理サーバは、接続ポイント識別子、並びにモビリティ端末システムの識別子、ユーザ名、ＰＩＤ等の他の関連情報を含む「ロケーション変更警告（Ｌｏｃａｔｉｏｎ　Ｃｈａｎｇｅ　Ａｌｅｒｔ）」を作成する。この警告は「ロケーション変更ＲＰＣ要求」内で使用された同じタイプ、長さ、データフォーマットでフォーマットされる。その後警告サブシステムが、警告のために登録された全てのアプリケーションにロケーション変更警告と共にこの情報を送る。警告のために登録されたアプリケーションは、現在の活動状況のモニタ、長期の動作ログ、ポリシー管理エンジン、第三者アプリケーション等の監視アプリケーション、並びにネットワーク管理ツールを含んでもよい。単一のそのような第三者アプリケーションは、このロケーション情報と、ウェブベースのマップとを組み合わせてモバイル端末システム又はＭＭＳの場所についての詳細情報を提供する。そのようなアプリケーションに加えて、他の動作をロケーション変更警告と関連付けることができる。これには、電子メールの送信、メッセージの印刷、プログラムの起動、および／またはポリシー変更が含まれる。

【０２４５】

ロケーション変更ＲＰＣはそのヘッダにロケーション変更、距離変更、又は速度変更によってトリガされたかどうかを示すフィールドを含む。

【０２４６】

ある例では、ＭＥＳは自身が移動したことを知らない場合がある。ＭＥＳが接続する媒体やネットワーク・アダプタに応じて、ＭＭＳはＭＥＳが新しい接続ポイントに渡ったことを知らせる唯一のエンティティであってもよい。モバイルルータの場合を考えてみる。ルータ以降のアドレスは同じままで、ルータのアドレスだけが変わる。この場合、ＭＭＳはＭＥＳのアドレスを新しく管理することを認識する。したがって、動作の検出を完了するためには、行動検出するためのＭＥＳとＭＭＳの両方の組み合わせの動作の検出が必要となる。本実施の形態では、ソースアドレスが変更され、新しいＩＭＰメッセージが受け取られる時に、ＭＭＳはＩＭＰ層でクライアントの行動を検出する。このことが行われると、ＭＭＳは局所的にロケーション変更警告を生成する。また、ＭＭＳは接続ポイントが変更したというメッセージをＭＥＳに送る。

【０２４７】

（トポロジＲＰＣの例）

「トポロジＲＰＣ要求（Ｔｏｐｏｌｏｇｙ　ＲＰＣ　Ｒｅｑｕｅｓｔ）」はモビリティ管理サーバからモバイル端末システムへ送られる。このＲＰＣの受け取りと同時に、モバイル端末システムはそのローカルデータ記憶装置に記憶されるトポロジ情報を読み出して、トポロジＲＰＣ応答（Ｔｏｐｏｌｏｇｙ　ＲＰＣ　Ｒｅｓｐｏｎｓｅ）を作成する。トポロジＲＰＣ応答は、前後との接続のタイプ、長さ情報が後に続く全長フィールド（Ｔｏｔａｌ　Ｌｅｎｇｔｈ　Ｆｉｅｌｄ）、タイプ、長さ情報が後に続く接続ポイント識別子、および、サブネットおよびネットワーク情報を示す値のデータによってフォーマットされる。この情報は、サーバで提供されるモバイルネットワークの完全なトポロジカル・マップを作成するのにサーバ上で使用されてもよい。

【０２４８】

（ロケーション情報ＵＩの例）

サーバ上のユーザインターフェイスは、ロケーション情報を対応付けして表示するための方法を提供する。このロケーション情報は、各アクティブなモバイル端末システムが利用でき、長期の動作ログは全てのアクティブなモバイル端末システム及び以前アクティブだったモバイル端末システムのロケーション変更履歴を保持する。ユーザインターフェイスによって、システム管理者は接続ポイント情報を人間が読める形式に構成できる。例えば、接続ポイント情報が４８ビットのＩＥＥＥ　ＭＡＣアドレスの形式で与えられると、サーバ上のユーザインターフェイスを介して提供される情報と共にこのＭＡＣアドレスが表示される。接続ポイントが「ホールマークカード（ＨａｌｌＭａｒｋ　Ｃａｒｄｓ）」店

のアクセスポイントを示すと、次の情報「ホールマーク、住所、市、州、郵便番号」を提示するように設定される。この情報がユーザに対して表示されることによって、「ホールマーク、住所、市、州、郵便番号」情報が提供されることになる。

【０２４９】

（ロケーションＲＰＣタイマーの例）

設定可能なタイマーがモバイル端末システムに設けられており、ロケーション変更ＲＰＣがモバイル端末システムからモビリティ管理サーバに送信される速度を制限する。タイマー間隔が接続ポイントの変更が起こる速度よりも大きい場合、モバイル端末システムはタイマー間隔が終了するまで待ってから別のロケーション変更ＲＰＣを生成する。

【０２５０】

（距離変更通知の例）

距離メトリクスは、ロケーション変更ＲＰＣの生成をトリガするために設けられる。この設定によって、ユーザが前にいた元のポイントから、ｎフィート、ｎキロメータ毎、あるいは他の適当な測定単位で３次元的に移動する時に、更新を送信するようにシステムが構成される。デフォルトによってこの設定は無効になる。この設定を可能にすることによって、設定された距離間隔を上回った時に変更通知が出される。

【０２５１】

（速度閾値通知の例）

速度変更メトリクスは、ロケーション変更ＲＰＣの生成をトリガするために設けられる。このパラメータは、一時間当たりのマイル等、一秒当たりの距離で構成される。このパラメータは、上限及び下限、並びに到達した速度を持続する必要のある時間間隔（すなわち、１０分間で０ＭＰＨ、又は１分間で７０ＭＰＨ）を特定する。このスピードに達すると、ロケーション変更通知が生成される。

【０２５２】

［応用例］

本発明は、実社会の様々な場面で応用される。例を以下に示す。

【０２５３】

（断続的に接続される携帯用コンピュータ）

多くの企業では、在宅勤務や、自宅で仕事をする社員がいることがある。そういった社員は、多くの場合、ラップトップコンピュータを使用して仕事をする。作業中に、社員は一般に自分達のラップトップコンピュータをドッキングポート又は他のコネクタを使用してイーサネット（Ｅｔｈｅｒｎｅｔ）等のローカルエリア・ネットワーク（Ｌｏｃａｌ ａｒｅａ ｎｅｔｗｏｒｋ）に接続する。ＬＡＮ接続によってネットワークサービス（例えば、プリンタ、ネットワークドライブ）やネットワーク・アプリケーション（例えば、データベースアクセス、電子データサービス）にアクセスが可能となる。

【０２５４】

あるプロジェクトに取り組む社員が、晩に帰宅する必要があり、自宅で仕事を続行したいとする。この社員は、ラップトップコンピュータで実行しているオペレーティングシステムとアプリケーションを「サスペンド」して、ラップトップコンピュータを片付け、そのラップトップコンピュータを自宅に持って帰ることができる。

【０２５５】

自宅に戻るとすぐに、その社員はラップトップコンピュータで実行しているオペレーティングシステムとアプリケーションを「レジューム」して、ダイアルアップ接続を介して、および／またはインターネットを通じてオフィスＬＡＮと再接続する。（ラップトップコンピュータが一時的に中断された間、ネットワークに対するラップトップコンピュータとそのアプリケーションのプロキシを継続した）モビリティ管理サーバは、ラップトップコンピュータを再認証し、ラップトップコンピュータとの通信を再開する。

【０２５６】

自宅で仕事をしている社員の立場から見ると、ネットワーク・ドライブ・マッピング、印刷サービス、電子メールセッション、データベースクエリ、他のネットワークサービスや

アプリケーションは全て会社で終わった状態そのままである。さらに、モビリティ管理サーバはラップトップコンピュータのセッションのプロキシを継続したため、社員の会社から自宅までの帰宅途中に、いずれのネットワーク・アプリケーションもラップトップコンピュータのセッションを終了しなかった。このように、本発明は、上記のような状況や他の状況において、高性能且つ有用な同一又は複数のネットワーク媒体を通じて効果的にセッションを持続可能とする。

【０２５７】

（モバイル在庫管理及び倉庫への応用）

大きな倉庫又は小売店チェーンを想定する。この構内において、在庫管理を行う従業員がパーソナルラップトップコンピュータ、並びにハンドヘルドデータ収集ユニットや端末装置などを搭載した乗物（すなわち、トラックやフォークリフト）を使って商品の在庫管理が行われる。

【０２５８】

倉庫や小売店の従業員は、ネットワークサブネットのわからない、管理監督の必要な不慣れなコンピュータユーザであることが多い。本発明によって倉庫ユーザがモバイルネットワークの複雑さを感じない、即使用可能なシステムの構築が可能となる。倉庫ユーザは、アクセスポイントの範囲内及び範囲外を動いたり、自分たちのモバイル端末システム１０４の中断及び再開をしたり、ホストセッション、ネットワークアドレス、又はトランスポート接続を気にせずに、位置を変えたりできる。さらに、モビリティ管理サーバ１０２の管理ソフトウェアは、従業員の生産能力を測るのに利用される処理数等のメトリクスを管理関係者に提供する。また、管理によりネットワークサブネット及びアクセスポイントを使用して従業員が最後にいたとされる物理的な位置を判定できる。

【０２５９】

（モバイル医療への応用）

無線ＬＡＮ技術を利用して幾つかの建物間でネットワーク通信を行う大きな病院を想定する。各建物は独自のサブネット上にある。本発明によって看護士や医師がハンドヘルドパーソナルコンピュータや端末を持って病室から病室を動き、病院のデータベースから患者情報の読出しや書込みが可能となる。ローカルデータベースやワールド・ワイド・ウェブ（Ｗｏｒｌｄ　Ｗｉｄｅ　Ｗｅｂ）を通じて、薬物治療や医療処置に関する最新記事へのアクセスが即座に可能となる。本発明はモバイル端末システム１０４への連続的な接続が可能なので、病院にいる間（一方向又は双方向の）ポケットベルはもはや必要なくなる。メッセージはモバイル端末システム１０４を介して医療関係者に直接送られる。倉庫の従業員の場合と同様、医療関係者は自分たちが利用しているモバイルネットワークについてわかる必要はない。さらに、モバイル端末システム１０４によって医療関係者は無線送信が望ましくないとされるエリア内（例えば、電波の放射によって医療機器に障害が出る可能性のある場所）での無線伝送ができなくなるが、中断したところから容易に再開して再接続可能となる。

【０２６０】

（トラック輸送及び貨物輸送）

運送会社は、貨物の状況を把握するのに本発明を用いる。倉庫に入っている間は、モバイル端末システム１０４はＬＡＮ技術を使用して倉庫内の貨物数を更新する。ローカルサービスから離れている間は、モバイル端末システム１０４はＣＤＰＤやＡＲＤＩＳ等のＷｉｄｅ　Ａｒｅａ　ＷＡＮサービスを使用してリアルタイムで貨物の状況や位置を維持できる。モバイル端末システム１０４はネットワーク・インフラストラクチャ間を自動的に切り替えるので、輸送関係者にネットワーク・トポロジの複雑さを感じさせない。

モバイル企業

企業の社員は、８０２．１１．等のインフラストラクチャが設けられた企業構内にいる間、本発明によるシステムを使用して、電子メール、ウェブコンテント、メッセージサービスへのアクセスを行う。ポケットベルサービスや他のモバイル機器のサービスはもはや必要ないので、これらを維持するためのコストが削減される。既存のモバイル機器サービス

の多くが提供する、コストのかかる「利用回数制（ｐａｙ－ｐｅｒ－ｕｓｅ）」モデルに対し、モバイルインフラストラクチャは一度の資金支出で購入できる。

【０２６１】

（ＩＰマルチプリケーション）

ある組織がインターネットに接続する必要のあるＬＡＮを有する場合、ＬＡＮの管理者には二つの選択肢がある。一つ目の選択は、ＬＡＮ上の全てのコンピュータに対して割り当てるためのグローバルアドレスを取得することであり、二つ目の選択は、グローバルアドレスをいくつか取得して、アドレス・マルチプライヤとして本発明に従ったモビリティ管理サーバ１０２を使用することである。多数のＩＰアドレスを取得することは、高額であるか不可能であるかである場合が多い。インターネット・サービス・プロバイダ（Ｉｎｔｅｒｎｅｔ　Ｓｅｒｖｉｃｅ　Ｐｒｉｖｉｄｅｒ；　ＩＳＰ）を利用してインターネットにアクセスする小規模の企業は、ＩＳＰが割り当てるＩＰアドレスだけを使用することになる。ＩＰアドレスの数は同時にインターネットに接続できるコンピュータの数を制限する。また、ＩＳＰは一回の接続につき課金するため、インターネット接続が必要なコンピュータが増えれば増えるほど、この方法はそれだけ費用がかかる。

【０２６２】

アドレス・マルチプライヤとして本発明に従ったモビリティ管理サーバ１０２を用いれば、これらの問題の多くを解決することができる。企業は、ＩＳＰを介してインターネットに接続するためのハードウェアとしてモビリティ管理サーバ１０２を用いることができる。これにより、モバイル端末システム１０４は容易に接続することができる。インターネットへの接続は全てモビリティ管理サーバ１０２を通じて行われるため、一つのアドレスだけをＩＳＰから取得すればよい。このように、アドレス・マルチプライヤとして本発明を用いることによって、企業はわずかな数（多くの場合一つ）のアドレスとアカウントをＩＳＰから取得すればよくなり、全体のＬＡＮでインターネットへの同時接続（十分な帯域幅が備えられていることを前提として）が可能になる。

【０２６３】

以上現在最も実用的且つ好適な実施形態とされる例と共に本発明を説明したが、本発明は開示した実施形態に限定されず、添付クレームの範囲内で種々に変形、同等に構成できるものとする。

【図面の簡単な説明】

【図１】

本発明のモバイル・コンピューティング・ネットワークの全体図である。

【図２】

モバイル端末システムとモビリティ管理サーバとのソフトウェア・アーキテクチャを例示するものである。

【図２Ａ】

モバイル端末システムとモビリティ管理サーバとの間で情報伝達を実行するステップを例示している。

【図３】

モバイル・インターセプタ・アーキテクチャを例示している。

【図３Ａ】

モバイル・インターセプタによって実行されるステップを例示したフローチャートである。

【図３Ｂ】

ＲＰＣ作業要求を扱うＲＰＣエンジン（ＲＰＣ　ｅｎｇｉｎｅ）によって実行されるステップを例示したフローチャートである。

【図４】

ＲＰＣ作業要求を処理するステップを例示したフローチャートである。

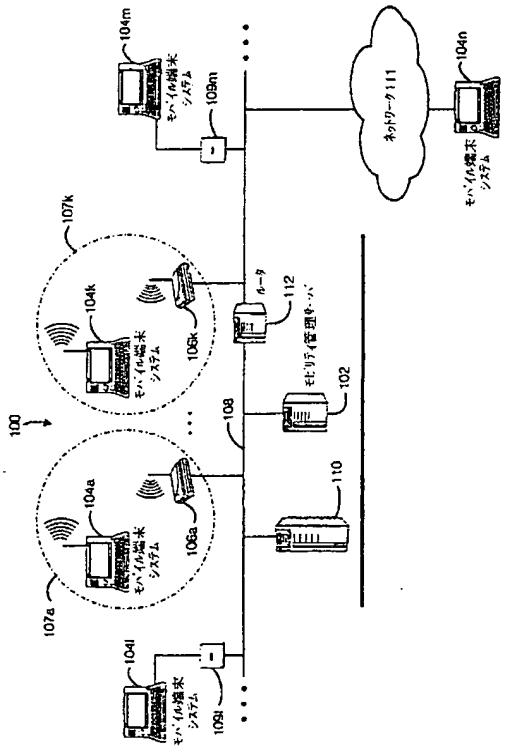
【図５】

ＲＰＣ作業要求を処理するステップを例示したフローチャートである。

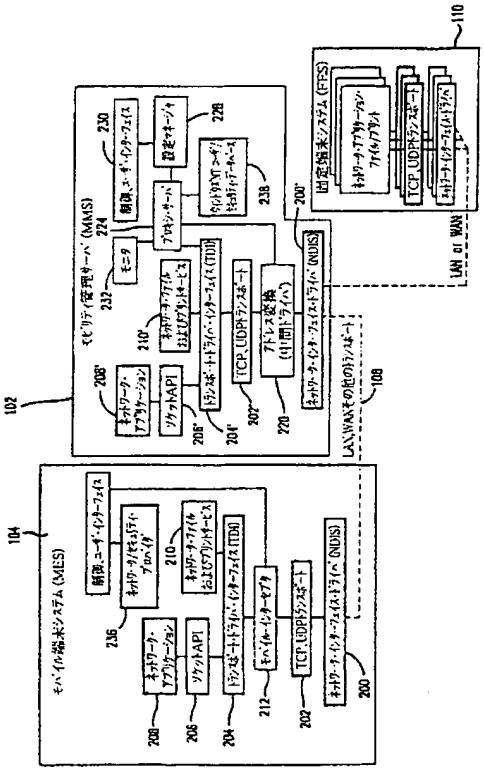
【図５Ａ】
ＲＰＣ作業要求を処理するステップを例示したフローチャートである。
【図５Ｂ】
ＲＰＣ作業要求を処理するステップを例示したフローチャートである。
【図５Ｃ】
ＲＰＣ作業要求を処理するステップを例示したフローチャートである。
【図６】
受信された作業要求を例示している。
【図７】
受信された作業要求が、どのようにそれぞれ別の優先度のキューにディスパッチされるかを示すものである。
【図８】
上記それぞれ別の優先度のキューにおけるコンテンツの処理を示している。
【図９】
上記それぞれ別の優先度のキューにおけるコンテンツの処理を示している。
【図１０Ａ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１０Ｂ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１０Ｃ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１１】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１２】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１２Ａ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１２Ｂ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１２Ｃ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１３Ａ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１３Ｂ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１３Ｃ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１４Ａ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示し

ている。
【図１４Ｂ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１５Ａ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１５Ｂ】
インターネット・モビリティ・プロトコルを提供するために実行されるステップを例示している。
【図１６】
リスナのデータ構造を例示している。
【図１７】
モバイル相互接続ローミング（ｍｏｂｉｌｅ　ｉｎｔｅｒｃｏｎｎｅｃｔ　ｒｏａｍｉｎｇ）を提供するためのステップを例示している。
【図１７Ａ】
モバイル相互接続ローミング（ｍｏｂｉｌｅ　ｉｎｔｅｒｃｏｎｎｅｃｔ　ｒｏａｍｉｎｇ）を提供するためのステップを例示している。
【図１８】
モバイル相互接続ローミング（ｍｏｂｉｌｅ　ｉｎｔｅｒｃｏｎｎｅｃｔ　ｒｏａｍｉｎｇ）を提供するためのステップを例示している。
【図１９Ａ】
インターフェイスによって補助されたローミング処理を例示する一つのフローチャートを形成している。
【図１９Ｂ】
インターフェイスによって補助されたローミング処理を例示する一つのフローチャートを形成している。
【図２０】
インターフェイスによって補助されているローミングのトポロジ・ノード・データ構造を例示する。
【図２１】
モビリティ管理システムのネットワークアドレスを、非接続ネットワーク・トポロジにおいてモバイル端末システムに配布する技術を例示している。
【図２２】
図２１の技術を、セキュアな通信を実現するために利用した例を示している。
【図２３】
図２１の技術を、分散型ネットワーク・インターフェイス・シナリオにおけるネットワークアドレスの変換に用いた例を示している。
【図２４】
ポリシー管理テーブルを例示している。
【図２５】
ポリシー管理処理のステップを例示したフローチャートである。

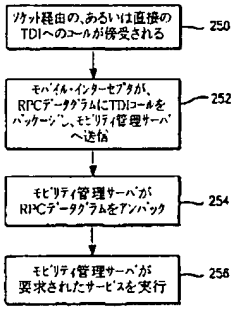
【図１】



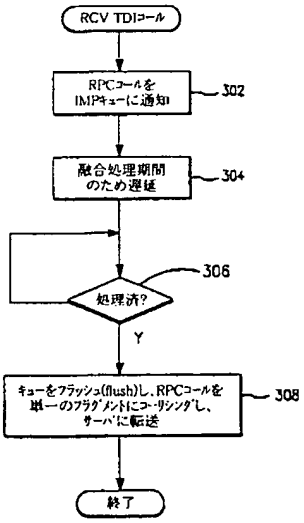
【図２】



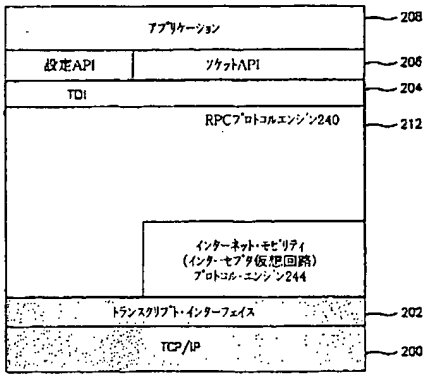
【図２Ａ】



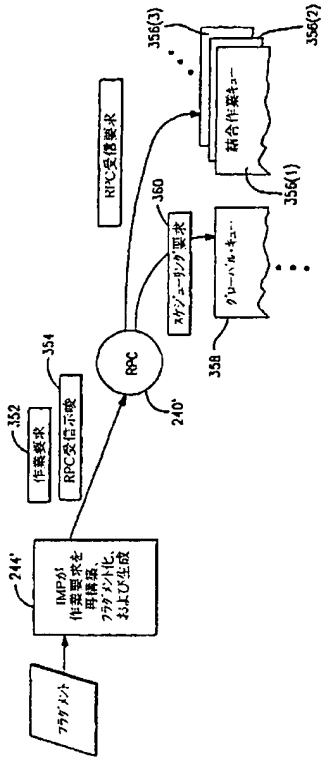
【図３Ａ】



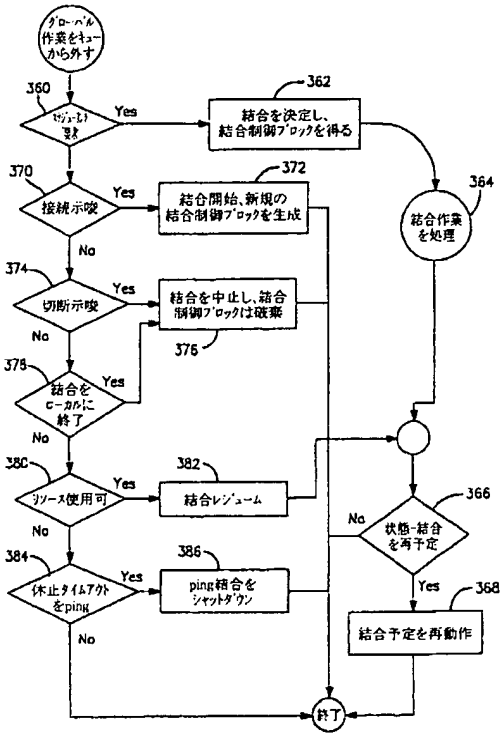
【図３】



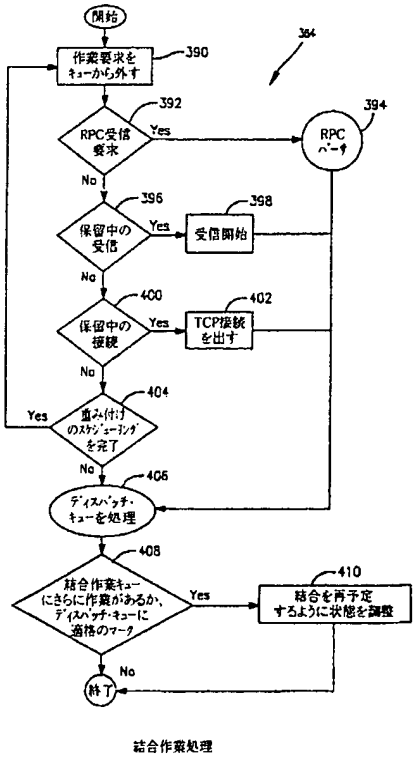
【図３Ｂ】



【図４】

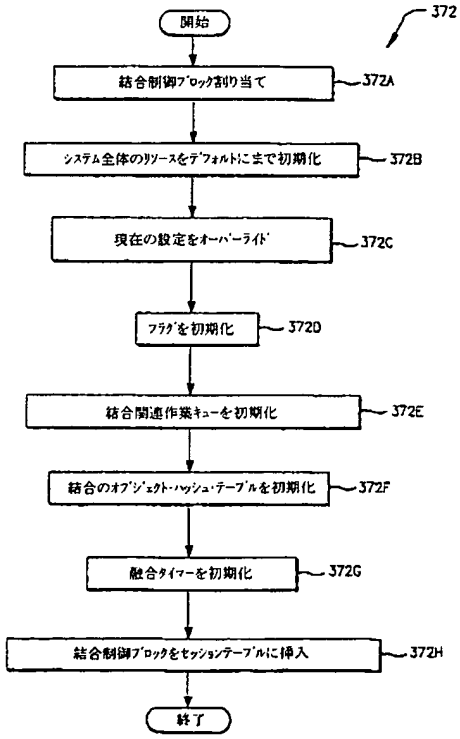


【図５】

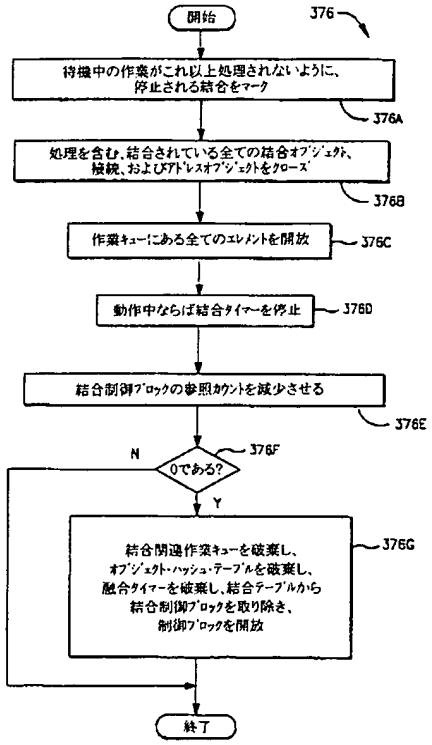


結合作業処理

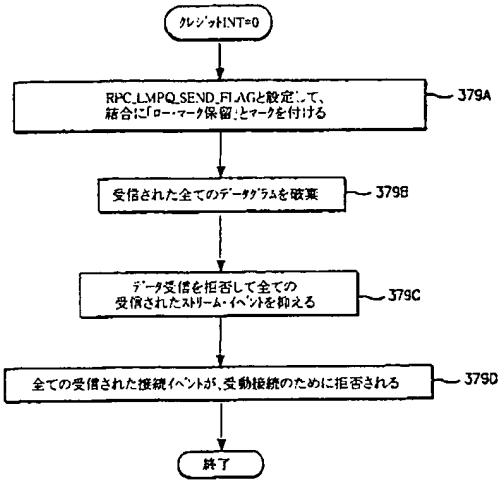
【図５Ａ】



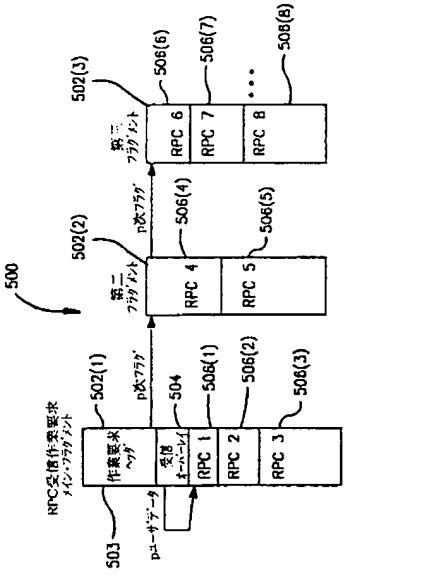
【図５Ｂ】


開始
376
待機中の作業がこれ以上処理されないように、
停止される結合をマーク
376A
処理を含む、結合されている全ての結合オブジェクト、
接続、およびアドレスオブジェクトをクローズ
376B
作業キューにある全てのエレメントを開放
376C
動作中ならば結合タイマーを停止
376D
結合制御ブロックの参照カウントを減少させる
376E
N
0である?
376F
Y
結合開通作業キューを破棄し、
オブジェクト・ハッシュ・テーブルを破棄し、
融合タイマーを破棄し、結合テーブルから
結合制御ブロックを取り除き、
制御ブロックを開放
376G
終了

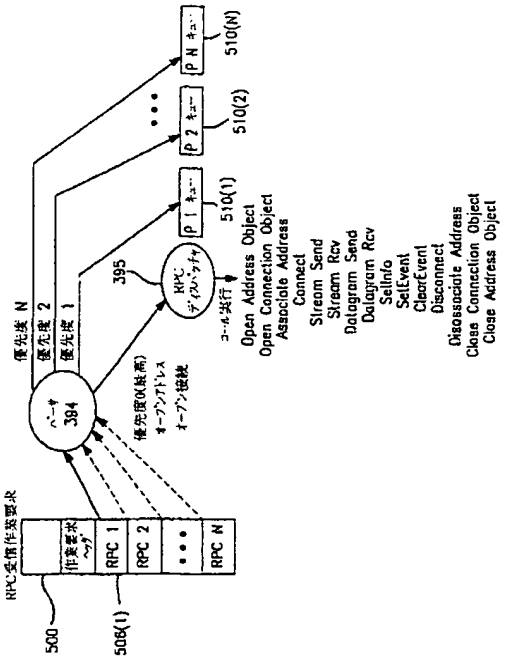
【図５Ｃ】


クレジットINT=0
RPC_LMPQ_SEND_FLAGと設定して、
結合に「ロー・マーク保留」とマークを付ける
379A
受信された全てのデータグラムを破棄
379B
データ受信を拒否して全ての
受信されたストリーム・イベントを抑える
379C
全ての受信された接続イベントが、受動接続のために拒否される
379D
終了

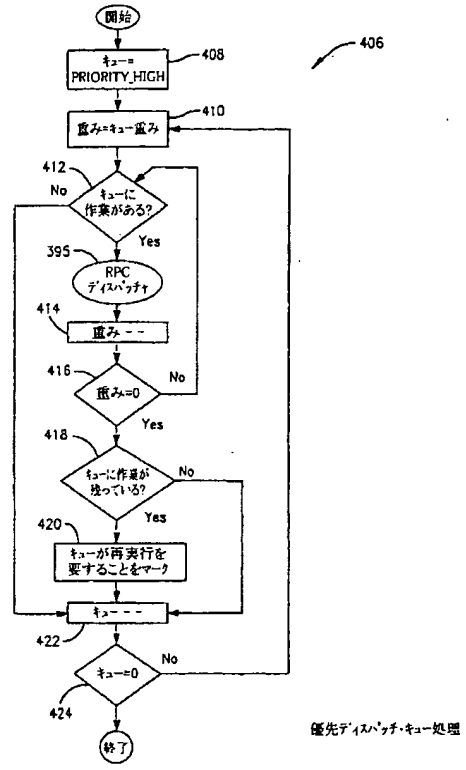
【図６】


500
502(1)
502(2)
502(3)
503
504
506(1)
506(2)
506(3)
506(4)
506(5)
506(6)
506(7)
506(8)
RPC 1
RPC 2
RPC 3
RPC 4
RPC 5
RPC 6
RPC 7
RPC 8

【図７】

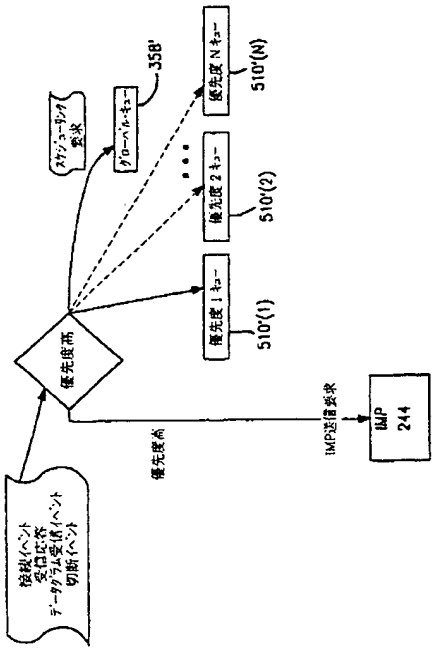

500
506(1)
510(1)
510(2)
510(N)
395
384
優先度 N
優先度 2
優先度 1
RPC 1
RPC 2
RPC N
Open Address Object
Open Connection Object
Associate Address
Connect
Stream Send
Stream Rcv
Datagram Send
Datagram Rcv
SetInfo
SetEvent
ClearEvent
Disconnect
Disassociate Address
Close Connection Object
Close Address Object

【図８】

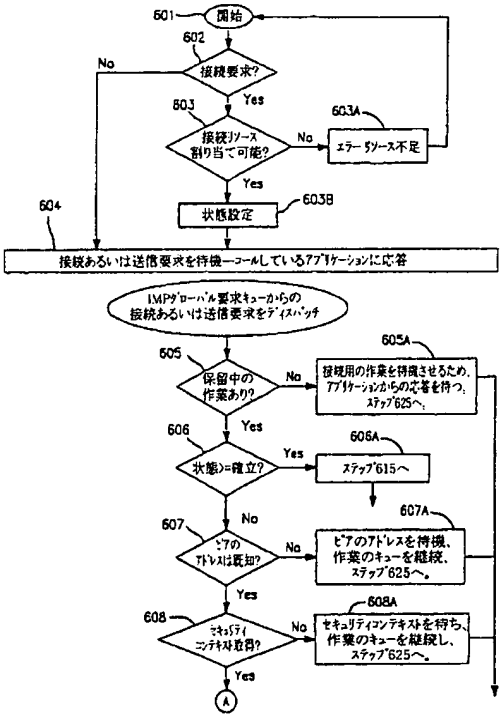


優先ディスパッチ・キュー処理

【図９】

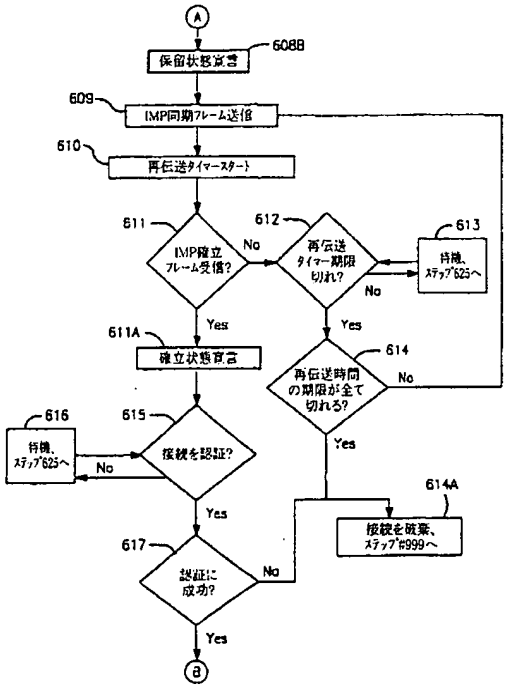


【図１０Ａ】

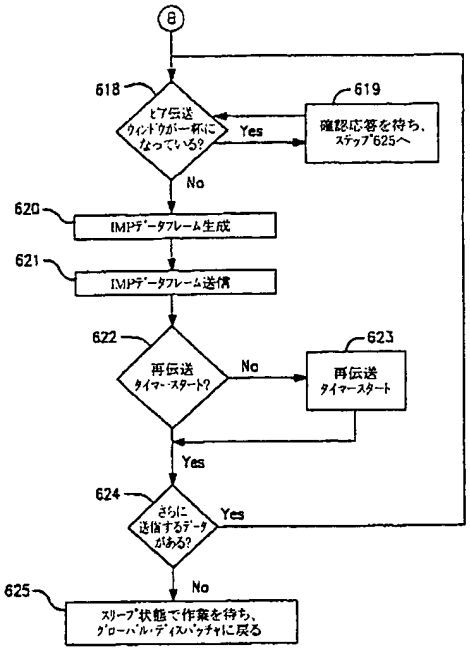


インターネット・モビリティ・プロトコルの接続決定ツリー
接続送信要求論理

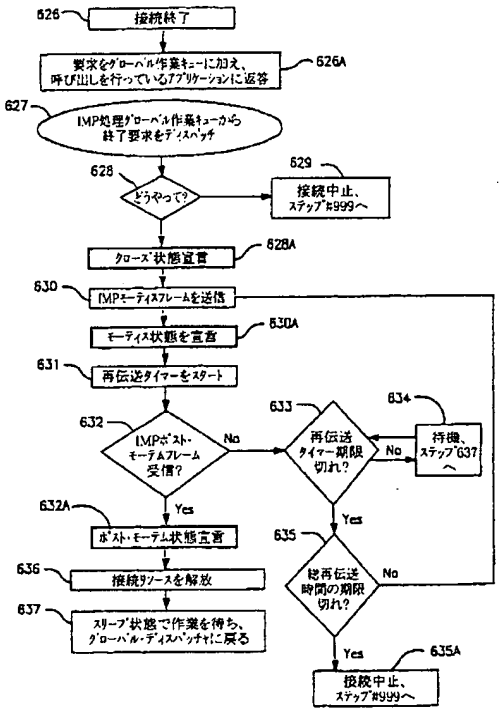
【図１０Ｂ】



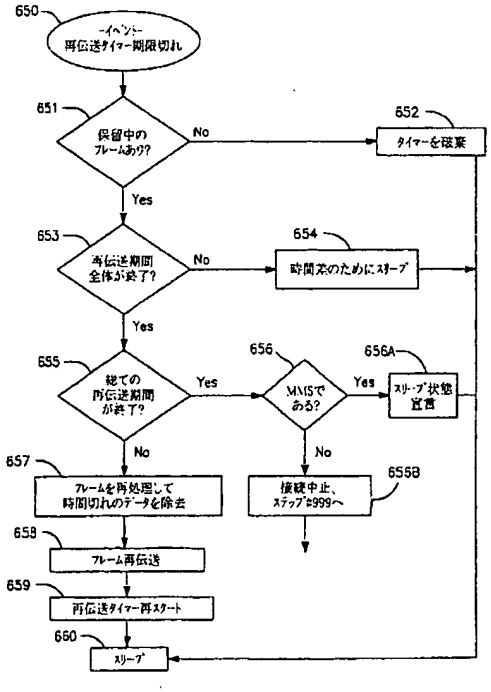
【図１０Ｃ】



【図１１】

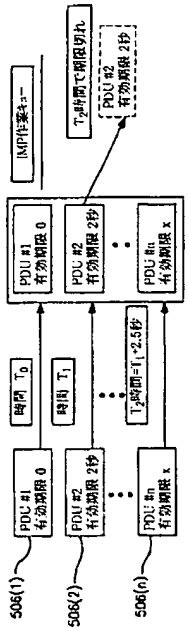


【図１２】

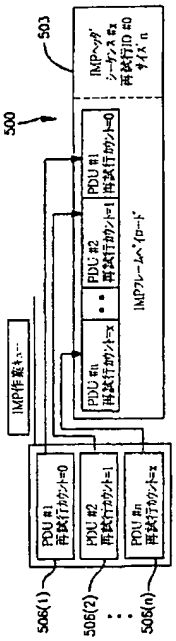


再伝送イベント論理

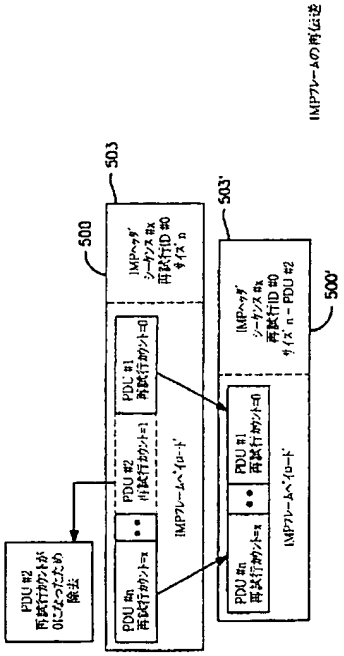
【図１２Ａ】



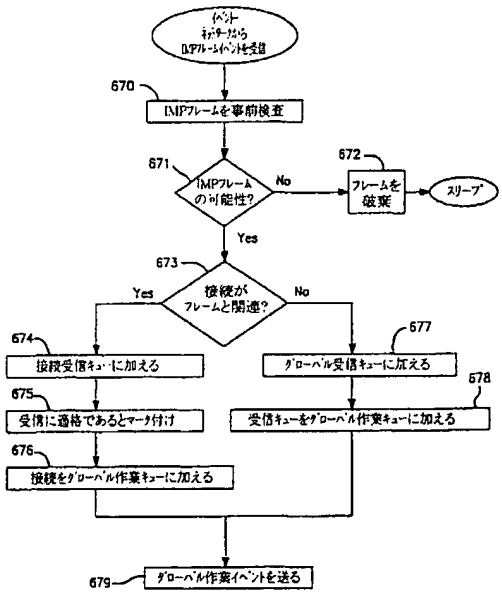
【図１２Ｂ】



【図１２Ｃ】

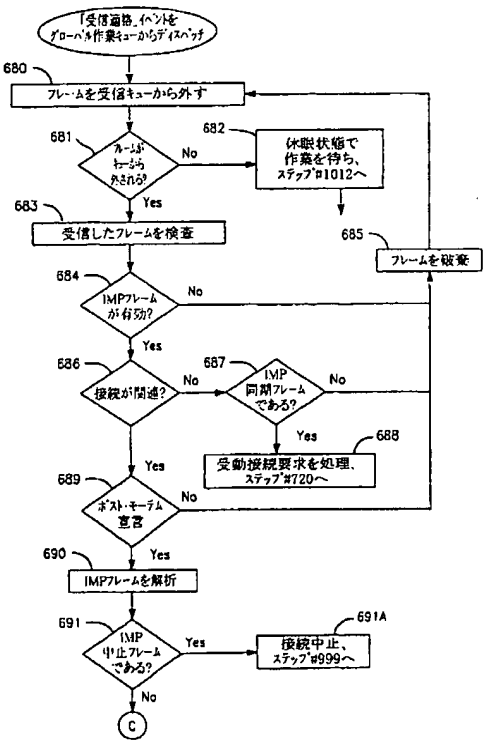


【図１３Ａ】

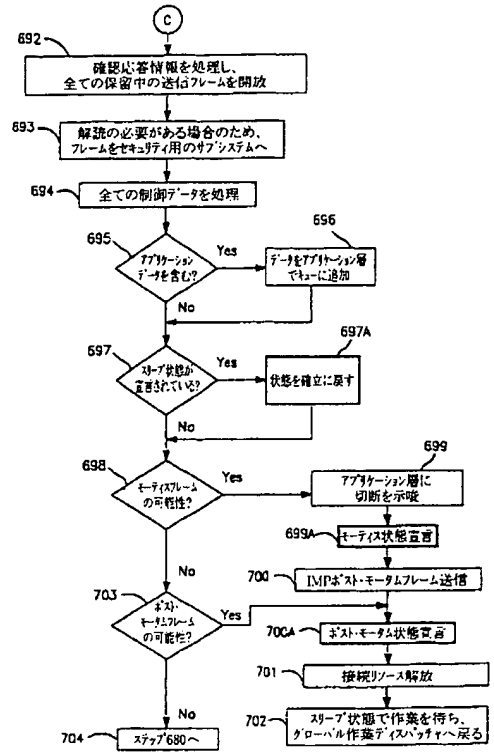


受信イベントロジック

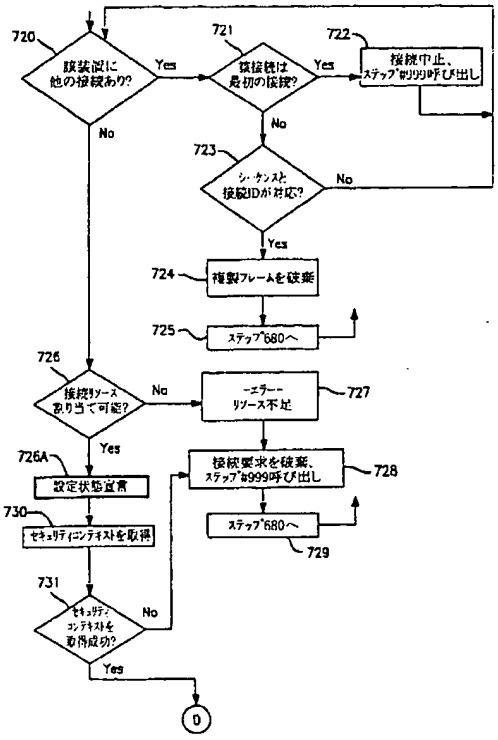
【図１３Ｂ】



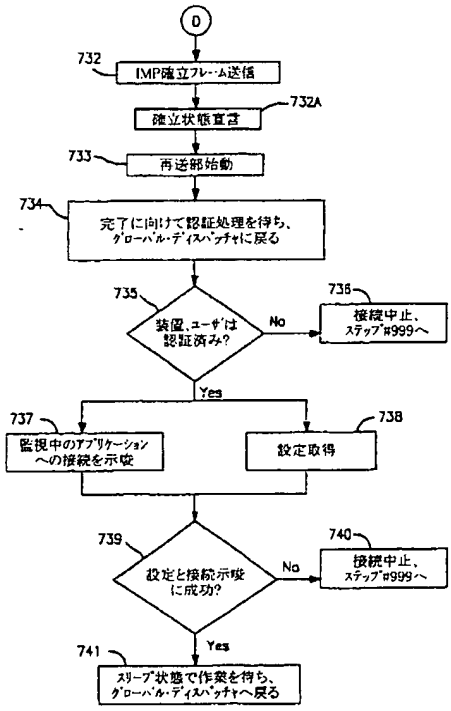
【図１３Ｃ】



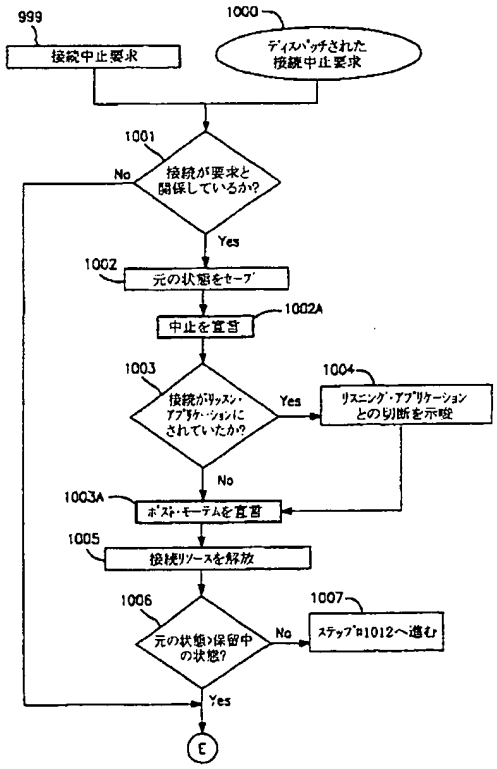
【図１４Ａ】



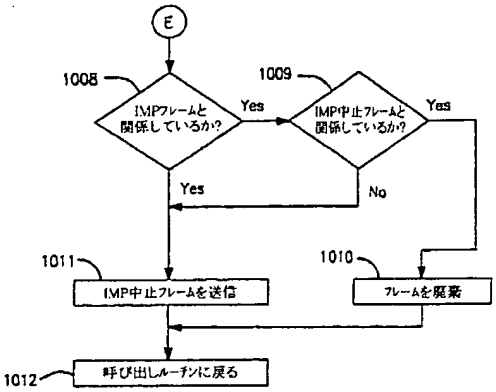
【図１４Ｂ】



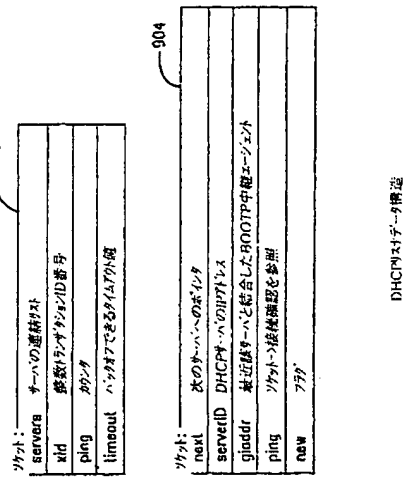
【図１５Ａ】



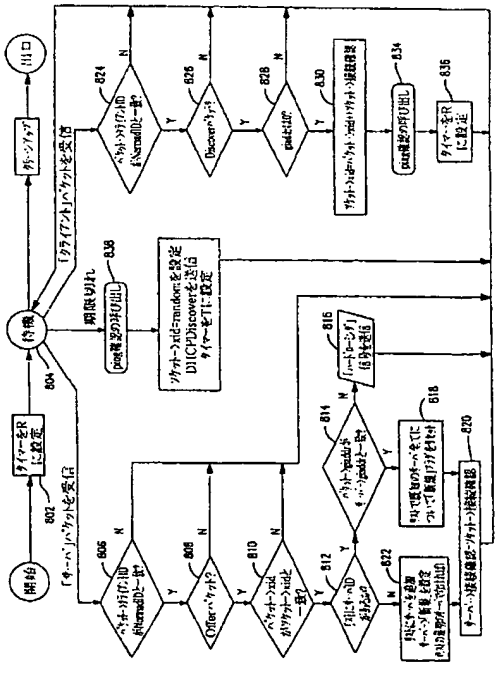
【図１５Ｂ】



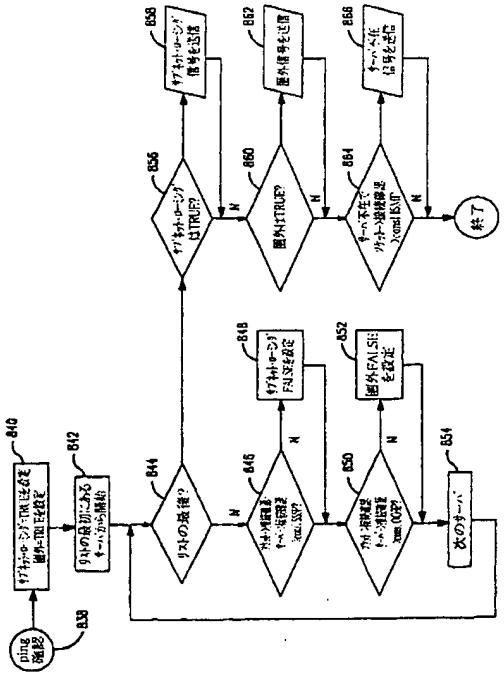
【図１６】



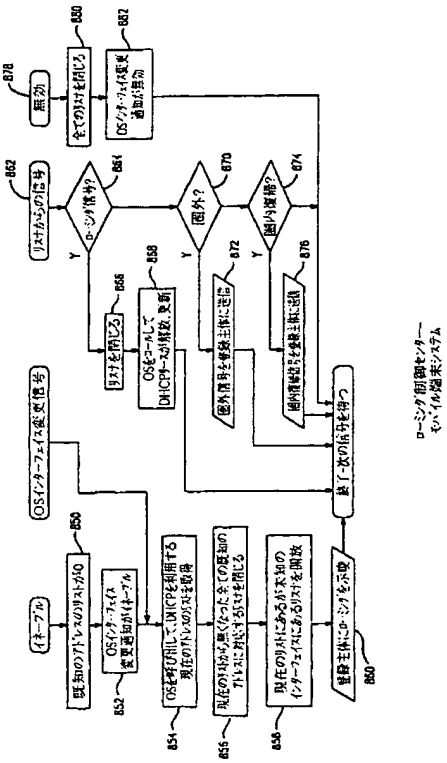
【図１７】



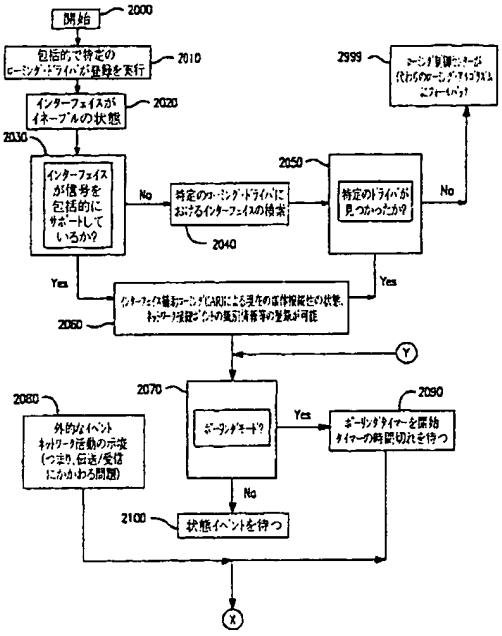
【図１７Ａ】



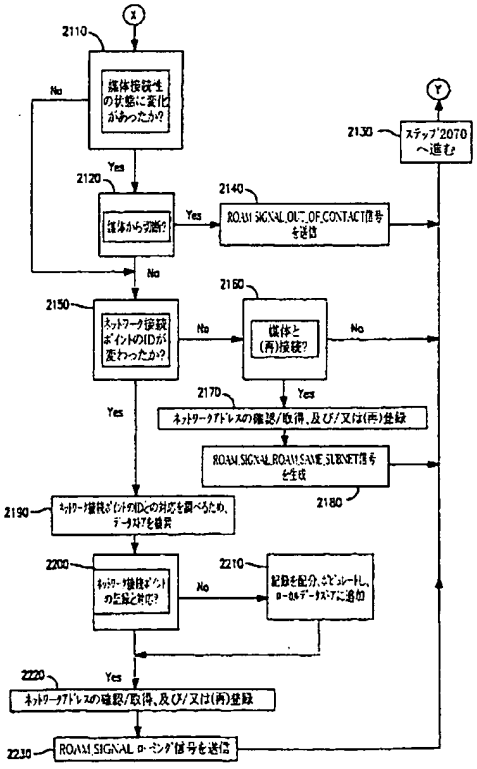
【図１８】



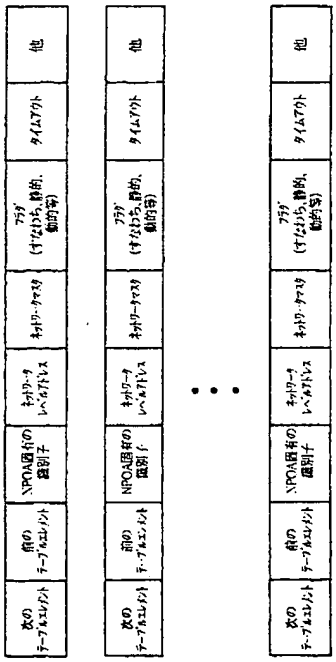
【図１９Ａ】



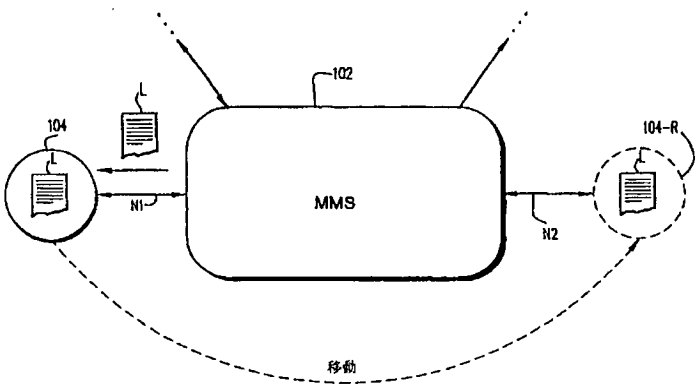
【図１９Ｂ】



【図２０】

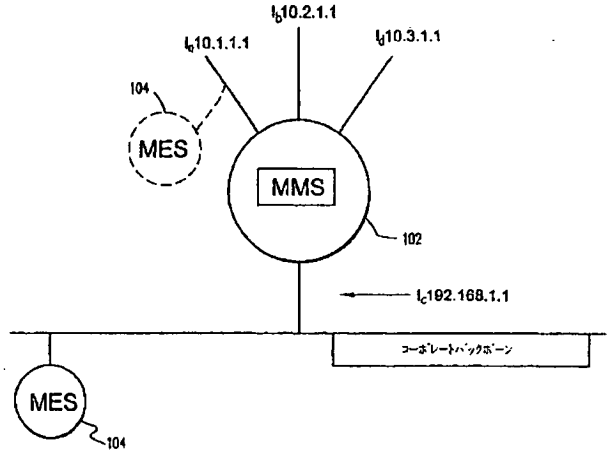


【図２１】



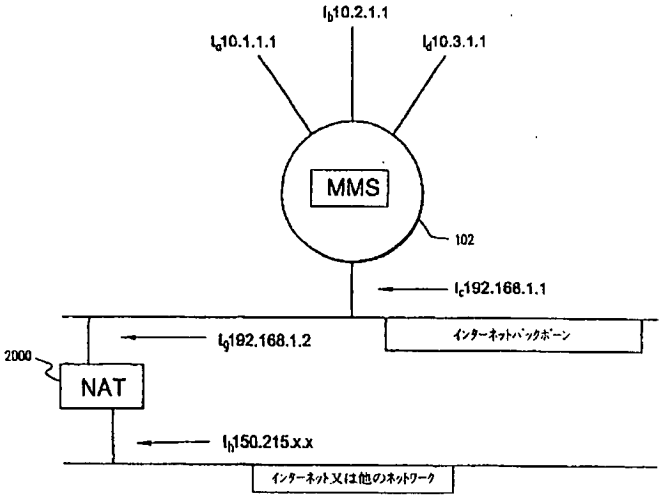
非接続ネットワークローミング

【図２２】



セキュアな非接続ネットワーク協調の例

【図２３】



【図２４】

ポリシー管理ルールテーブル例

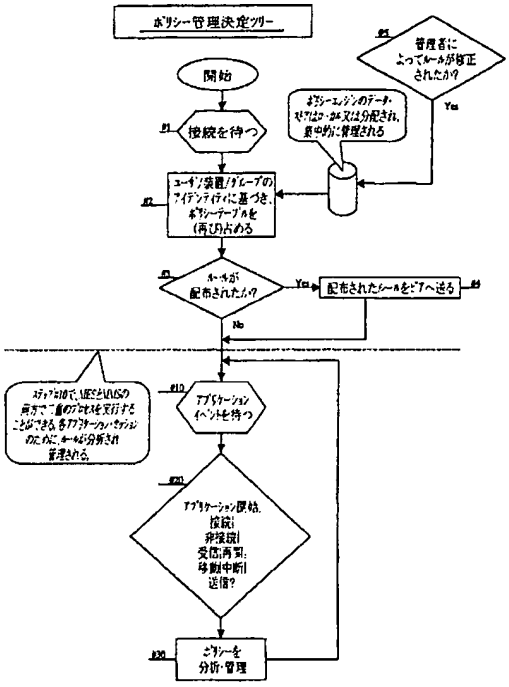
通信先/受信	プロトコル	MESのソースポート	MESのソースアドレス	MESの宛先ポート	MESの目的地アドレス	帯域幅（利用可能量）	プロセス名	ネットワーク	ロケーション（GPS座標）	ネットワークの接続ポイント	ユーザ	特定要求
送信/受	Yes	任意	任意	21	任意	< 100,000	任意	任意	任意	任意	米国特許庁	Yes
送信/受	Yes	任意	任意	20	任意	< 100,000	任意	任意	任意	任意	米国特許庁	Yes
送信	No	5008	任意	5008	10.1.1.1		任意	任意	任意	任意	米国特許庁	No
受信	No	5008	10.1.1.1	5008	任意		任意	任意	任意	任意	米国特許庁	No

前提条件

1. ピア・ファイル・トランスファ・プロトコル制御及びデータポートは21と20
2. *はワイルドカード
3. MMSのネットワークアドレス及びポートは10.1.1.1と5008
4. MMSからフレームを受け取るMESのネットワークポートは5008

上記例で、利用可能な帯域幅が一秒間で100,000バイト以下に低減される場合、目的地ポート20,21との上記接続は全て否定され、減速される。この例では、ルール(列)3,4によってネットワークトラフィックのみがMMSへ、又はMMSから流れることが可能となる。プロキシが行われない他の全てのネットワークトラフィックは暗に廃棄される。このテーブルはポリシー管理で否定できるメトリクス一式を表すものではない。他に、金額的なコスト、位置、ネットワークの接続ポイント等の変数が意思決定の樹状図に追加することができる。これらの項目を解釈するルールエンジンがMESとMMS間に配布することができる。そういうものとして、どちらかの側、あるいは両方が指定のポリシーを実施してもよい。

【図２５】


ポリシー管理決定ツリー
開始
接続を待つ
ユーザ/装置/グループの
アイデンティティに基づき、
ポリシーテーブルを
(再び)占める
ルールが
配布されたか?
Yes
No
配布されたルールをピアへ送る
アプリケーション
イベントを待つ
アプリケーション開始,
接続,
非接続,
受信再試,
移動中断,
送信?
ポリシーを
分析・管理
管理者に
よってルールが修正
されたか?
Yes

【国際公開パンフレット】

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 March 2002 (21.03.2002)

PCT

(10) International Publication Number
WO 02/23362 A1

(51) International Patent Classification⁷: G06F 15/16

(21) International Application Number: PCT/US01/28191

(22) International Filing Date:
12 September 2001 (12.09.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/660,300 12 September 2000 (12.09.2000) US
60/274,615 12 March 2001 (12.03.2001) US

(71) Applicant (*for all designated States except US*): NETMOTION WIRELESS, INC. [US/US]; 1500 Dexter Avenue North, Seattle, WA 98109-3032 (US).

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): HANSON, Aaron, D. [US/US]; 3002 NW 63rd Street, Seattle, WA 98107 (US). STURNIOLO, Emil, A. [US/US]; 4050 Alameda Court, Medina, OH 44256 (US). BIENN, Austroj [US/US]; 816 North 175th, #4, Seattle, WA 98133 (US). OLSON, Erik, D. [US/US]; 306 NW 82nd Street, Seattle, WA 98117 (US). SAVARESE, Joseph, T. [US/US]; 22205 95th Place West, Edmonds, WA 98020 (US).

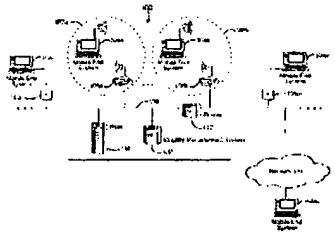
(74) Agent: PARIS, Robert, W.; Nixon & Vanderhye, 1100 North Glebe Road, Suite 800, Arlington, VA 22201-4714 (US).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR PROVIDING MOBILE AND OTHER INTERMITTENT CONNECTIVITY IN A COMPUTING ENVIRONMENT



(57) Abstract: A seamless solution transparently addresses the characteristics of nomadic systems, and enables existing network applications to run reliably in mobile environments. A Mobility Management Server (102) coupled to the mobile network maintains the state of each of any number of Mobile End Systems (104) and handles the complex session management required to maintain persistent connections to the network and to other peer processes. If a Mobile End System becomes unreachable, suspends, or changes network address (e.g., due to roaming from one network interconnect to another), the Mobility Management Server maintains the connection to the associated peer task - allowing the Mobile End System to maintain a continuous connection even though it may temporarily lose contact with its network medium. An interface-based listener uses network point of attachment information supplied by a network interface to determine roaming conditions and to efficiently establish connection upon roaming. The Mobility Management Server can distribute lists to Mobile End Systems specifying how to contact it over disjoint networks.

WO 02/23362 A1

WO 02/23362 A1

Published:
— with international search report
before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND APPARATUS FOR PROVIDING MOBILE AND OTHER INTERMITTENT CONNECTIVITY IN A COMPUTING ENVIRONMENT

FIELD OF THE INVENTION

The present invention relates to connectivity between networked computing devices. More particularly, the present invention relates to methods and systems that transparently address the characteristics of nomadic systems, and enable existing network applications to run reliably in the associated mobile environments. Still more particularly, the invention relates to techniques and systems for providing a continuous data stream connection between intermittently-connected devices such as handheld data units and personal computing devices.

BACKGROUND AND SUMMARY OF THE INVENTION

Increasingly, companies are seeing rapid access to key information as the way to maintaining a competitive advantage. To provide immediate access to this information, mobile and other intermittently-connected computing devices are quietly and swiftly becoming an essential part of corporate networks -- especially with the proliferation of inexpensive laptops and hand-held computing devices. However, integrating these nomadic devices into existing network infrastructures has created a challenge for the information manager.

Many problems in mobile networking parallel the difficulties in early local area networks (LANs) before the adoption of Ethernet. There are a variety of mobile protocols and interfaces, and because standards are just developing, there is little interoperability between systems. In addition, performance over these network technologies has typically been slow and

bandwidth limited. Implementation costs to date have been high due the specialized nature of deployed systems.

Along with these issues, mobile technologies present a category of problems unto their own. Interconnects back into the main network may travel over and through a public network infrastructure, thus allowing sensitive information to possibly be tapped into. Furthermore, if any of the intermediary interconnects are via a wireless interface, the information is actually broadcast, and anyone with a similar interface can eavesdrop without much difficulty.

But, perhaps even more significantly, mobile networking has generally in the past been limited to mostly message-oriented or stateless applications -- and thus has not been readily adaptable for existing or new corporate applications that use client/server, host-terminal, web-based or shared file systems models. This is because such commonly used applications need stateful sessions that employ a continuous stream of data -- not just a stateless packet exchange -- to work effectively and reliably.

To this end, many or most popular off-the-shelf networking applications require TCP/IP sessions, or private virtual circuits. These sessions cannot continue to function if they encounter network interruptions, nor can they tolerate roaming between networks (i.e., a change of network addresses) while established. Yet, mobile networking is, by its nature, dynamic and unreliable. Consider these common scenarios encountered in mobile networks:

Disconnected or Out of Range User

When a mobile device disconnects from a given network or loses contact (e.g., through an outage or "hole" in the coverage of a wireless interconnect), the session-oriented application running on the mobile device

loses its stateful connection with its peer and ceases to operate. When the device is reattached or moves back into contact, the user must re-connect, log in again for security purposes, find the place in the application where work was left off, and possibly re-enter lost data. This reconnection process is time consuming, costly, and can be very frustrating.

Moving to a Different Network or Across a Router Boundary (Network Address Change)

Mobile networks are generally segmented for manageability purposes. But the intent of mobile devices is to allow them to roam. Roaming from one network interconnect to another can mean a change of network address. If this happens while the system is operational, the routing information must be changed for communications to continue between the associated peers. Furthermore, acquiring a new network address may require all of the previously established stateful application sessions to be terminated -- again presenting the reconnection problems noted above.

Security

As mentioned before, companies need to protect critical corporate data. Off-the-shelf enterprise applications are often written with the assumption that access to the physical network is controlled (i.e., carried within cables installed inside a secure facility), and security is maintained through an additional layer of authentication and possible encryption. These assumptions have not been true in the nomadic computing world – where data is at risk for interception as it travels over public airways or public wire-line infrastructures.

It would be highly desirable to provide an integrated solution that transparently addresses the characteristics of nomadic systems, and enables existing network applications to run reliably in these mobile environments.

The present invention solves this problem by providing a seamless solution that extends the enterprise network, letting network managers provide mobile users with easy access to the same applications as stationary users without sacrificing reliability or centralized management. The solution combines advantages of present-day wire-line network standards with emerging mobile standards to create a solution that works with existing network applications.

In accordance with one aspect of a non-limiting exemplary and illustrative embodiment of the present invention, a Mobility Management Server (MMS) coupled to the mobile interconnect maintains the state of each of any number of Mobile End Systems (MES) and handles the complex session management required to maintain persistent connections to the network and to peer application processes. If a Mobile End System becomes unreachable, suspends, or changes network address (e.g., due to roaming from one network interconnect to another), the Mobility Management Server maintains the connection to the associated peer -- allowing the Mobile End System to maintain a continuous virtual connection even though it may temporarily lose its actual physical connection.

The illustrative non-limiting example embodiments provided by the present invention also provide the following (among others) new and advantageous techniques and arrangements:

- a Mobility Management Server providing user configurable session priorities for mobile clients;

- per-user mobile policy management for managing consumption of network resources;
- a roaming methodology making use of the industry standard Dynamic Host Configuration Protocol (DHCP) in coordination with a Mobility Management Server;
- automatic system removal of unreliable datagrams based on user configurable timeouts; and
- automatic system removal of unreliable datagrams based on user configurable retries.

In more detail, the preferred illustrative embodiments of the present invention in one of their aspects provide a Mobility Management Server that is coupled to the mobile interconnect (network). The Mobility Management Server maintains the state of each of any number of Mobile End Systems and handles the complex session management required to maintain persistent connections to the network and to other processes (e.g., running on other network-based peer systems). If a Mobile End System becomes unreachable, suspends, or changes network address (e.g., due to roaming from one network interconnect to another), the Mobility Management Server maintains the connection to the associated peer, by acknowledging receipt of data and queuing requests. This proxying by the Mobility Management Server allows the application on the Mobile End System to maintain a continuous connection even though it may temporarily lose its physical connection to a specific network medium.

In accordance with another aspect of preferred embodiments of the present invention, a Mobility Management Server manages addresses for Mobile End Systems. Each Mobile End System is provided with a proxy address on the primary network. This highly available address is known as the "virtual address" of the Mobile End System. The Mobility Management

Server maps the virtual addresses to current "point of presence" addresses of the nomadic systems. While the point of presence address of a Mobile End System may change when the mobile system changes from one network interconnect to another, the virtual address stays constant while any connections are active or longer if the address is statically assigned.

In accordance with yet another aspect of the preferred exemplary embodiments of the present invention, a Mobility Management Server provides centralized system management of Mobile End Systems through a console application and exhaustive metrics. The preferred embodiment also provides user configurable session priorities for mobile clients running through a proxy server, and per-user mobile policy management for managing consumption of network resources.

In accordance with yet another aspect, a Remote Procedure Call protocol and an Internet Mobility Protocol are used to establish communications between the proxy server and each Mobile End System.

Remote procedure calls provide a method for allowing a process on a local system to invoke a procedure on a remote system. The use of the RPC protocol allows Mobile End Systems to disconnect, go out of range or suspend operation without losing active network sessions. Since session maintenance does not depend on a customized application, off-the-shelf applications will run without modification in the nomadic environment.

The Remote Procedure Call protocol generates transactions into messages that can be sent via the standard network transport protocol and infrastructure. These RPC messages contain the entire network transaction initiated by an application running on the Mobile End System -- enabling the Mobility Management Server and Mobile End System to keep connection state information synchronized at all times -- even during interruptions of the physical link connecting the two. In the preferred

embodiment of the present invention providing RPC's, the proxy server and the Mobile End Systems share sufficient knowledge of each transaction's state to maintain coherent logical database about all shared connections at all times.

The Internet Mobility Protocol provided in accordance with the preferred embodiment of the present invention compensates for differences between wired local area network interconnects and other less reliable networks such as a wireless LAN or WAN. Adjusted frame sizes and protocol timing provide significant performance improvements over non-mobile-aware transports -- dramatically reducing network traffic. This is important when bandwidth is limited or when battery life is a concern. The Internet Mobility Protocol provided in accordance with the preferred embodiment of the present invention also ensures the security of organizational data as it passes between the Mobile End System and the Mobility Management Server over public network interconnects or airways. The Internet Mobility Protocol provides a basic firewall function by allowing only authenticated devices access to the organizational network. The Internet Mobility Protocol can also certify and encrypt all communications between the Mobility Management Server and the Mobile End System.

In accordance with yet another aspect of preferred non-limiting embodiments of the present invention, mobile inter-connectivity is built on standard transport protocols (e.g., TCP/IP, UDP/IP and DHCP, etc) to extend the reach of standard network application interfaces. The preferred exemplary embodiments of the present invention efficiently integrates transport, security, address management, device management and user management needs to make nomadic computing environments effectively transparent. The Internet Mobility Protocol provides an efficient

mechanism for multiplexing multiple streams of data (reliable and unreliable) through a single virtual channel provided by such standard transport protocols over standard network infrastructure.

With the help of the RPC layer, the Internet Mobility Protocol coalesces data from different sources targeted for the same or different destinations, together into a single stream and forwards it over a mobile link. At the other end of the mobile link, the data is demultiplexed back into multiple distinct streams, which are sent on to their ultimate destination(s). The multiplexing/demultiplexing technique allows for maximum use of available bandwidth (by generating the maximum sized network frames possible), and allows multiple channels to be established (thus allowing prioritization and possibly providing a guaranteed quality of service if the underlying network provides the service).

The Internet Mobility Protocol provided in accordance with the preferred example embodiments of the present invention provide the additional non-limiting exemplary features and advantages:

- Transport protocol independence.
- Allows the network point of presence (POP) or network infrastructure to change without affecting the flow of data (except where physical boundary, policy or limitations of bandwidth may apply).
- Minimal additional overhead.
- Automatic fragment resizing to accommodate the transmission medium. (When the protocol data unit for a given frame is greater then the available maximum transmission unit of the network medium, the Internet Mobility Protocol will fragment and reassemble the frame to insure that it can traverse the network. In the event of a retransmit, the frame will again be

assessed. If the network infrastructure or environment changes, the frame will be refragmented or in the case that the maximum transmission unit actually grew, sent as a single frame.)

- Semantics of unreliable data are preserved, by allowing frames to discard unreliable data during retransmit.
- Provides a new semantic of Reliable Datagram service. (Delivery of datagrams can now be guaranteed to the peer terminus of the Internet Mobility Protocol connection. Notification of delivery can be provided to a requesting entity.)
- Considers the send and receive transmission path separately, and automatically tailors its operating parameters to provided optimum throughput. (Based on hysteresis, it adjusts such parameters as frame size/fragmentation threshold, number of frames outstanding (window), retransmit time, and delayed acknowledgement time to reduce the amount of duplicate data sent through the network.)
- Network fault tolerant (since the expected usage is in a mobile environment, temporary loss of network medium connectivity does not result in a termination of the virtual channel or application based connection).
- Provides an in-band signaling method to its peer to adjust operating parameters (each end of the connection can alert its peer to any changes in network topology or environment).
- Employs congestion avoidance algorithms and gracefully decays throughput when necessary.
- Employs selective acknowledgement and fast retransmit policies to limit the number of gratuitous retransmissions, and provide faster handoff recovery in nomadic environments. (This also

allows the protocol to maintain optimum throughput in a lossy network environment.)

- Employs sliding window technology to allow multiple frames to be outstanding. (This parameter is adjustable in each direction and provides for streaming frames up to a specified limit without requiring an acknowledgement from its peer.)
- Sequence numbers are not byte oriented, thus allowing for a single sequence number to represent up to a maximum payload size.
- Security aware. (Allows for authentication layer and encryption layer to be added in at the Internet Mobility Protocol layer.)
- Compression to allow for better efficiency through bandwidth limited links.
- Balanced design, allowing either peer to migrate to a new point of presence.
- Either side may establish a connection to the peer.
- Allows for inactivity timeouts to be invoked to readily discard dormant connections and recover expended resources.
- Allows for a maximum lifetime of a given connection (e.g., to allow termination and/or refusal to accept connections after a given period or time of day).

Non-limiting preferred exemplary embodiments of the present invention also allow a system administrator to manage consumption of network resources. For example, the system administrator can place controls on Mobile End Systems, the Mobility Management Server, or both. Such controls can be for the purpose, for example, of managing allocation of network bandwidth or other resources, or they may be related to security

issues. It may be most efficient to perform management tasks at the client side for clients with lots of resources. However, thin clients don't have many resources to spare, so it may not be practical to burden them with additional code and processes for performing policy management. Accordingly, it may be most practical to perform or share such policy management functions for thin clients at a centralized point such as the Mobility Management Server. Since the Mobility Management Server proxies the distinct data streams of the Mobile End Systems, it provides a central point from which to conduct policy management. Moreover, the Mobility Management Server provides the opportunity to perform policy management of Mobile End Systems on a per user and/or per device basis. Since the Mobility Management Server is proxying on a per user basis, it has the ability to control and limit each user's access to network resources on a per-user basis as well as on a per-device basis.

As one simple example, the Mobility Management Server can "lock out" certain users from accessing certain network resources. This is especially important considering that interface network is via a mobile interconnect, and may thus "extend" outside of the boundaries of a locked organizational facility (consider, for example, an ex-employee who tries to access the network from outside his former employer's building). However, the policy management provided by the Mobility Management Server can be much more sophisticated. For example, it is possible for the Mobility Management Server to control particular Web URL's particular users can visit, filter data returned by network services requests, and/or compress data for network bandwidth conservation. This provides a way to enhance existing and new application-level services in a seamless and transparent manner.

Furthermore, because the Mobile End System may be connected to an "untrusted" network (i.e. outside the corporations locked boundaries) there is a chance of malicious attack while being remotely connected. By sharing policy rules and filters with the Mobile End System, one can protect the MES from rogue connections, provide ingress filtering for the remote node, and further secure the corporate infrastructure from one central location.

A further exemplary embodiment of the invention provides an interface-assisted roaming listener that allows Mobile End Systems to take advantage of interfaces supporting generic signaling, to enable interface-assisted roaming. In accordance with one feature provided in accordance with the exemplary embodiment, the Mobile End System interface-based listener determines in response to an event (e.g., a callback, a timer timeout or a network activity hint indicating data loss), whether the Mobile End System's media connectivity status has changed. For example, the listener signals to the interface when it detects that the Mobile End System has become detached and is no longer in communication with the network. Upon re-attachment, the listener uses previously recorded network point of attachment identification information to determine whether the Mobile End System has been reattached to the same or different network point of attachment. If the reattachment is to the same network point of attachment, the listener signals to alert the mobile clients that they need to take steps to reestablish transport level communications. If the reattachment is to a different network point of attachment, the listener signals a "roam" condition and prompts the Mobile End System to acquire an address that is usable on the current network segment (this may entail, for example, registering the current address to be valid on a new subnet, for example). The listener may maintain a network topology map (which may be learned

through operation) to assist it in deciding the correct signal (e.g., "roam same subnet" or "roam") to generate to its clients.

A still further aspect provided by non-limiting preferred exemplary embodiments of our invention allows access to a Mobility Management Server (MMS) in a "disjoint networking" mode. The new algorithm allows for dynamic/static discovery of alternate network addresses that can be used to establish/continue communications with an MMS – even in a disjoint network topology in which one network infrastructure may have no knowledge of network addresses for another network infrastructure. In accordance with this arrangement, a list of alternate addresses that the MMS is available at is preconfigured, forwarded to or dynamically learned by an MES (Mobile End System) during the course of a conversation/connection. In one embodiment, the MMS can use a connection over one network to send the MES one or more MMS network addresses or other MMS identities corresponding to other networks. This list can be sent/updated during circuit creation or at any other time during the connection.

If/when the MES roams to a second network, it uses the list of MMS "alias" addresses/identifications to contact the MMS over the new network connection on the second network. This allows the MES to re-establish contact with the MMS over the new network connection even though the first and second networks may not share any addresses, routes, or other information.

Further example embodiments of the invention provide policy management decision making within a distributed mobile network topology. For example, rule-based policy management procedures can be performed to allow, deny and/or condition request fulfillment based on a variety of metrics. Such policy management can be used, for example, to provide

decision making based on cost metrics such as least cost routing in a multi-home/path environment.

Such policy management techniques may take into account the issue of mobility or positioning (i.e., roaming) in making decisions. For example, policy management rules may be based on locale of the device (e.g., proximity to network point of attachment such as access point/base station, hubs, routers, or GPS coordinate) so certain operations can be allowed in one building of an enterprise but not in another building. An example of such an application might be an enterprise with several different wireless networks. Such an enterprise might have a loading dock and office area served by a wireless network. The system administrator would be able to configure the system such that workers (e.g., users and devices) on the loading dock are not permitted access to the wireless network in the office environment. Also policy management results can be tri-state: allow, deny or delay (for example, if the decision is based on bandwidth requirements or cost, the decision may be to delay an operation from being executed and to wait for a more opportune time when the operation can be accommodated).

The policy management provided by the preferred example embodiments is capable of modifying an operation based on a decision. For example, one example action is to throttle consumption of network bandwidth for all active applications. Also consider an aggressive application that is consuming significant bandwidth. The policy engine can govern the rate at which the application's operations/transactions are completed. This behavior may also be learned dynamically to squelch a possible errant application. Another example action provides reconstitution of data(i.e. dithering of graphics images based on available/allowable bandwidth or cost/user restrictions).

Furthermore the rules engine allows for other actions to be invoked based on rule evaluation. External procedures such as logging an event, sending an alert or notifying the user that the action is being denied, delayed, or conditioned may be executed. Such notification can also be interactive and ask for possible overrides to an existing rule from the operator.

The policy management engine provided in the example non-limiting embodiment can base its decision making on any number or combination metrics that are associated with the device, device group, user group, user,/or network point of attachment.

As part of the policy management functionality, other locale base information and services may also be acquired/provided for the purposes of policy management, network modeling, and/or asset tracking. Such services include the ability to automatically present to users and mobile end systems information that is applicable within the context of a mobile end system's present location. This information may be provided in the form of messages, files, or in some other electronic format.

One non-limiting example of such use of this capability would permit shopping malls, business communities, and large retailers, to locate wireless access points that may be compatible with Bluetooth PANs, IEEE 802.11 LANs, 802.15 PANs, or other wireless network standards in strategic locations within the shopping center. As customers roam from location to location, stores and vendors would be permitted to push down information relevant to the vendors that are present within the mobile end systems current location. This information would include information such as current sales, discounts, and services. In addition to such information, mobile end systems may be provided electronic coupons used for sales promotion. Vendors would be permitted to register for these services

through some centralized authority that may be associated with the mall, business community, retailer, or some other hosted service.

A further example non-limiting use of such a technology would be in vertical industries where information is collected based on location including but not limited to such industries as field service, field sales, package delivery, or public safety where lists of local services, maps, directions, customers, or hazards may be pushed down to mobile end systems based on location.

Yet another non-limiting example use may entail a web based service for monitoring and tracking mobile end systems. For example, customers may register for this tracking service so trusted third parties may log onto the hosted web service and find out exact locations of their mobile end systems. This may include mobile end systems installed on vehicles or carried by pedestrians. As mobile end systems continue to experience reductions in size and wait, such services become more likely. These services would permit people to track and locate individuals that are high risk such as elderly, handicapped, or ill. It may also be used to track items that are highly valued such as automobiles or other expensive mobile properties and packages. Using 3G WAN networks, Bluetooth networks, 802.11 networks, 802.15 networks, and other wireless technologies, combined with this unique ability to provide seamless connectivity while switching network mediums or point of attachments, such services become possible and likely at a much reduced cost.

The present invention thus extends the enterprise network, letting network managers provide mobile users with easy access to the same applications as stationary users without sacrificing reliability or centralized management. The solution combines advantages of existing wire-line

network standards with emerging mobility standards to create a solution that works with existing network applications.

BRIEF DESCRIPTION OF THE DRAWINGS

These, as well as other features and advantages of this invention, will be more completely understood and appreciated by careful study of the following more detailed description of presently preferred example embodiments of the invention taken in conjunction with the accompanying drawings, of which:

Figure 1 is a diagram of an overall mobile computing network provided in accordance with the present invention;

Figure 2 shows an example software architecture for a Mobile End System and a Mobility Management Server;

Figure 2A shows example steps performed to transfer information between a Mobile End System and a Mobility Management Server;

Figure 3 shows an example mobile interceptor architecture;

Figure 3A is a flowchart of example steps performed by the mobile interceptor;

Figure 3B is a flowchart of example steps performed by an RPC engine to handle RPC work requests;

Figures 4-5C are flowcharts of example steps to process RPC work requests;

Figure 6 is a diagram of an example received work request;

Figure 7 is a diagram showing how a received work request can be dispatched onto different priority queues;

Figures 8 and 9 show processing of the contents of the different priority queues;

Figures 10A-15B show example steps performed to provide an Internet Mobility Protocol;

Figure 16 shows example listener data structures;

Figures 17, 17A and 18 are flowcharts of example steps performed to provide for mobile interconnect roaming;

Figures 19A and 19B are together a flowchart of an example interface-assisted roaming process;

Figure 20 shows an example interface assisted roaming topology node data structure;

Figure 21 shows an example technique for distributing mobility management system network addresses to mobile end systems in a disjoint network topology;

Figure 22 shows an example use of the Figure 21 technique to provide secure communications;

Figure 23 shows an example use of the Figure 21 technique to provide network address translation in a distributed network interface scenario;

Figure 24 shows an example policy management table; and

Figure 25 is a flowchart of example policy management processing steps

DETAILED DESCRIPTION OF NON-LIMITING PRESENTLY PREFERRED EXAMPLE EMBODIMENTS

Figure 1 is an example of mobile enhanced networked computer system 100 provided in accordance with the present invention. Networked computer system 100 includes a Mobility Management Server 102 and one or more Mobile End Systems 104. Mobile End Systems 104 can communicate with Mobility Management Server 102 via a local area

network (LAN) 108. Mobility Management Server 102 serves as network level proxy for Mobile End Systems 104 by maintaining the state of each Mobile End System, and by handling the complex session management required to maintain persistent connections to any peer systems 110 that host network applications – despite the interconnect between Mobile End Systems 104 and Mobility Management Server 102 being intermittent and unreliable. In the preferred embodiment, Mobility Management Server 102 communicates with Mobile End Systems 104 using Remote Procedure Call and Internet Mobility Protocols in accordance with the present invention.

In this particular example, Mobile End Systems 104 are sometimes but not always actively connected to Mobility Management Server 102. For example:

- Some Mobile End Systems 104a-104k may communicate with Mobility Management Server 102 via a mobile interconnect (wirelessly in this case), e.g., conventional electromagnetic (e.g., radio frequency) transceivers 106 coupled to wireless (or wire-line) local area or wide area network 108. Such mobile interconnect may allow Mobile End Systems 104a-104k to "roam" from one cover area 107a to another coverage area 107k. Typically, there is a temporary loss of communications when a Mobile End System 104 roams from one coverage area 107 to another, moves out of range of the closest transceiver 106, or has its signal temporarily obstructed (e.g., when temporarily moved behind a building column or the like).
- Other Mobile End Systems 104l, 104m, ... may communicate with Mobility Management Server 102 via non-permanent wire-based interconnects 109 such as docking ports, network cable connectors, or the like. There may be a temporary loss of communications when Mobile End Systems 104 are temporarily

disconnected from LAN 108 by breaking connection 109, powering off the Mobile End Systems, etc.

• Still other Mobile End Systems (e.g., 104n) may be nomadically coupled to Mobility Management Server 102 via a further network topography 111 such as a wide area network, a dial-up network, a satellite network, or the Internet, to name a few examples. In one example, network 111 may provide intermittent service. In another example, Mobile End Systems 104 may move from one type of connection to another (e.g., from being connected to Mobility Management Server 102 via wire-based interconnect 109 to being connected via network 111, or vice versa) -- its connection being temporarily broken during the time it is being moved from one connection to another.

Mobile End Systems 104 may be standard mobile devices and off the shelf computers. For example, Mobile End System 104 may comprise a laptop computer equipped with a conventional radio transceiver and/or network cards available from a number of manufacturers. Mobile End Systems 104 may run standard network applications and a standard operating system, and communicate on the transport layer using a conventionally available suite of transport level protocols (e.g., TCP/IP suite.) In accordance with the present invention, Mobile End Systems 104 also execute client software that enables them to communicate with Mobility Management Server 102 using Remote Procedure Call and Internet Mobility Protocols that are transported using the same such standard transport level protocols.

Mobility Management Server 102 may comprise software hosted by a conventional Windows NT or other server. In the preferred embodiment, Mobility Management Server 102 is a standards-compliant, client-server

based intelligent server that transparently extends the enterprise network 108 to a nomadic environment. Mobility Management Server 102 serves as network level proxy for each of any number of Mobile End Systems 104 by maintaining the state of each Mobile End System, and by handling the complex session management required to maintain persistent connections to any peer systems 110 that host network applications -- despite the mobile interconnect between Mobile End Systems 104 and transceivers 106 being intermittent and unreliable.

For example, server 102 allows any conventional (e.g., TCP/IP based) network application to operate without modification over mobile connection. Server 102 maintains the sessions of Mobile End Systems 104 that disconnect, go out of range or suspend operation, and resumes the sessions when the Mobile End System returns to service. When a Mobile End System 104 becomes unreachable, shuts down or changes its point of presence address, the Mobility Management Server 102 maintains the connection to the peer system 110 by acknowledging receipt of data and queuing requests until the Mobile End System once again becomes available and reachable.

Server 102 also extends the management capabilities of wired networks to mobile connections. Each network software layer operates independently of others, so the solution can be customized to the environment where it is deployed.

As one example, Mobility Management Server 102 may be attached to a conventional organizational network 108 such as a local area network or wide area network. Network 108 may be connected to a variety of fixed-end systems 110 (e.g., one or most host computers 110). Mobility Management Server 102 enables Mobile End Systems 104 to communicate with Fixed End System(s) 110 using continuous session type data streams

even though Mobile End Systems 104 sometimes lose contact with their associated network interconnect or move from one network interconnect 106, 109, 111 to another (e.g., in the case of wireless interconnect, by roaming from one wireless transceiver 106 coverage area 107 to another).

A Mobile End System 104 establishes an association with the Mobility Management Server 102, either at startup or when the Mobile End System requires network services. Once this association is established, the Mobile End System 104 can start one or more network application sessions, either serially or concurrently. The Mobile End System 104-to-Mobility Management Server 102 association allows the Mobile End System to maintain application sessions when the Mobile End System, disconnects, goes out of range or suspends operation, and resume sessions when the Mobile End System returns to service. In the preferred embodiment, this process is entirely automatic and does not require any intervention on the user's part.

In accordance with an aspect of the present invention, Mobile End Systems 104 communicate with Mobility Management Server 102 using conventional transport protocols such as, for example, UDP/IP. Use of conventional transport protocols allows Mobile End Systems 104 to communicate with Mobility Management Server 102 using the conventional routers 112 and other infrastructure already existing on organization's network 108. In accordance with the present invention, a higher-level Remote Procedure Call protocol generates transactions into messages that are sent over the mobile enhanced network 108 via the standard transport protocol(s). In this preferred embodiment, these mobile RPC messages contain the entire network transaction initiated by an application running on the Mobile End System 104, so it can be completed in its entirety by the Mobility Management Server. This enables the Mobility Management

Server 102 and Mobile End System 104 to keep connection state information synchronized at all times -- even during interruptions of network medium connectivity.

Each of Mobile End Systems 104 executes a mobility management software client that supplies the Mobile End System with the intelligence to intercept all network activity and relay it via the mobile RPC protocol to Mobility Management Server 102. In the preferred embodiment, the mobility management client works transparently with operating system features present on Mobile End Systems 104 (e.g., Windows NT, Windows 98, Windows 95, Windows CE, etc.) to keep client-site application sessions active when contact is lost with the network.

Mobility Management Server 102 maintains the state of each Mobile End System 104 and handles the complex session management required to maintain persistent connections to associated peer 108 such as host computer 110 attached to the other end of the connection end point. If a Mobile End System 104 becomes unreachable, suspends, or changes network address (e.g., due to roaming from one network interconnect to another), the Mobility Management Server 102 maintains the connection to the host system 110 or other connection end-point, by acknowledging receipt of data and queuing requests. This proxy function means that the peer application never detects that the physical connection to the Mobile End System 104 has been lost -- allowing the Mobile End System's application(s) to effectively maintain a continuous connection with its associated session end point (by simply and easily resuming operations once a physical connection again is established) despite the mobile system temporarily losing connection or roaming from one network interconnect 106A to another network interconnect 106K within coverage area 107K.

Mobility Management Server 102 also provides address management to solve the problem of Mobile End Systems 104 receiving different network addresses when they roam to different parts of the segmented network. Each Mobile End System 104 is provided with a virtual address on the primary network. Standard protocols or static assignment determine these virtual addresses. For each active Mobile End System 104, Mobility Management Server 102 maps the virtual address to the Mobile End System's current actual ("point of presence") address. While the point of presence address of a Mobile End System 104 may change when the device changes from one network segment to another, the virtual address stays constant while any connections are active or longer if the address is assigned statically.

Thus, the change of a point of presence address of a Mobile End System 104 remains entirely transparent to an associated session end point on host system 110 (or other peer) communicating with the Mobile End System via the Mobility Management Server 102. The peer 110 sees only the (unchanging) virtual address proxied by the server 102.

In the preferred embodiment, Mobility Management Server 102 can also provide centralized system management through console applications and exhaustive metrics. A system administrator can use these tools to configure and manage remote connections, and troubleshoot remote connection and system problems.

The proxy server function provided by Mobility Management Server 102 allows for different priority levels for network applications, users and machines. This is useful because each Mobility Management Server 102 is composed of finite processing resources. Allowing the system manager to configure the Mobility Management Server 102 in this way provides enhanced overall system and network performance. As one example, the

system manager can configure Mobility Management Server 102 to allow real time applications such as streaming audio or video to have greater access to the Mobility Management Server 102's resources than other less demanding applications such as email.

In more detail, Mobility Management Server 102 can be configured via an application or application interface; standard network management protocols such as SNMP; a Web-based configuration interface; or a local user interface. It is possible to configure association priority and/or to configure application priority within an association. For example, the priority of each association relative to other associations running through the Mobility Management Server 102 is configurable by either the user name, or machine name (in the preferred embodiment, when the priority is configured for both the user and the machine that a user is logged in on, the configuration for the user may have higher precedence). In addition or alternatively, each association may have several levels of application priority, which is configured based on network application name. The system allows for any number of priority levels to exist. In one particular implementation, three priority levels are provided: low, medium and high.

Server and Client Example Software Architecture

Figure 2 shows an example software architecture for Mobile End System 104 and Mobility Management Server 102. In accordance with one aspect of the present invention, Mobile End System 104 and Mobility Management Server 102 run standard operating system and application software -- with only a few new components being added to enable reliable and efficient persistent session connections over an intermittently connected mobile network 108. As shown in Figure 2, Mobile End System 104 runs conventional operating system software including network interface drivers

200, TCP/UDP transport support 202, a transport driver interface (TDI) 204, and a socket API 206 used to interface with one or more conventional network applications 208. Conventional network file and print services 210 may also be provided to communicate with conventional TDI 204. Server 102 may include similar conventional network interface drivers 200', TCP/UDP transport support 202', a transport driver interface (TDI) 204', and a socket API 206' used to interface with one or more conventional network applications 208'. Mobile End System 104 and Mobility Management Server 102 may each further include conventional security software such as a network/security provider 236 (Mobile End System) and a user/security database 238 (server).

In accordance with the present invention, a new, mobile interceptor component 212 is inserted between the TCP/UDP transport module 202 and the transport driver interface (TDI) 204 of the Mobile End System 104 software architecture. Mobile interceptor 212 intercepts certain calls at the TDI 204 interface and routes them via RPC and Internet Mobility Protocols and the standard TCP/UDP transport protocols 202 to Mobility Management Server 102 over network 108. Mobile interceptor 212 thus can intercept all network activity and relay it to server 102. Interceptor 212 works transparently with operating system features to allow client-side application sessions to remain active when the Mobile End System 104 loses contact with network 108.

While mobile interceptor 212 could operate at a different level than the transport driver interface 204 (e.g., at the socket API level 206), there are advantages in having mobile interceptor 212 operate at the TDI level or more specifically, any transport protocol interface. For brevity sake, all references to the transport driver interface will be denoted using the acronym TDI. Many conventional operating systems (e.g., Microsoft

Windows 95, Windows 98, Windows NT and Windows CE, etc.) provide TDI interface 204 -- thus providing compatibility without any need to change operating system components. Furthermore, because the transport driver interface 204 is normally a kernel level interface, there is no need to switch to user mode -- thus realizing performance improvements. Furthermore, mobile interceptor 212 working at the level of TDI interface 204 is able to intercept from a variety of different network applications 208 (e.g., multiple simultaneously running applications) as well as encompassing network file, print and other kernel mode services 210 (which would have to be handled differently if the interceptor operated at the socket API level 206 for example).

Figure 2A shows an example high level flowchart of how mobile interceptor 212 works. A call to the TDI interface 204 of Mobile End System 104 (block 250) is intercepted by mobile interceptor 212 (block 252). Mobile interceptor 212 packages the intercepted RPC call in a fragment in accordance with an Internet Mobility Protocol, and sends the fragment as a datagram via a conventional transport protocol such as UDP or TCP over the LAN, WAN or other transport 108 to Mobility Management Server 102 (block 252). The Mobility Management Server 102 receives and unpackages the RPC datagram (block 254), and provides the requested service (for example, acting as a proxy to the Mobile End System application 208 by passing data or a response to a application server process running on Fixed End System 110).

Referring once again to Figure 2, Mobility Management Server 102 includes an address translator 220 that intercepts messages to/from Mobile End Systems 104 via a conventional network interface driver 222. For example, address translator 230 recognizes messages from an associated session peer (Fixed End System 110) destined for the Mobile End System

104 virtual address. These incoming Mobile End System messages are provided to proxy server 224, which then maps the virtual address and message to previously queued transactions and then forwards the responses back to the current point of presence addresses being used by the associated Mobile End System 104.

As also shown in Figure 2, Mobility Management Server 102 includes, in addition to address translation (intermediate driver) 220, and proxy server 224, a configuration manager 228, a control/user interface 230 and a monitor 232. Configuration management 228 is used to provide configuration information and parameters to allow proxy server 224 to manage connections. Control, user interface 230 and monitor 232 allow a user to interact with proxy server 224.

Mobile Interceptor

Figure 3 shows an example software architecture for mobile interceptor 212 that support the RPC Protocol and the Internet Mobility Protocol in accordance with the present invention. In this example, mobile interceptor 212 has two functional components:

a Remote Procedure Call protocol engine 240; and

an Internet Mobility Protocol engine 244.

Proxy server 224 running on Mobility Management Server 102 provides corresponding engines 240', 244'.

Mobile interceptor 212 in the preferred embodiment thus supports Remote Procedure Call protocol and Internet Mobility Protocol to connect Mobility Management Server 102 to each Mobile End Systems 104. Remote procedure calls provide a method for allowing a process on a local system to invoke a procedure on a remote system. Typically, the local system is not aware that the procedure call is being executed on a remote

system. The use of RPC protocols allows Mobile End Systems 104 to go out of range or suspend operation without losing active network sessions. Since session maintenance does not depend on a customized application, off-the-shelf applications will run without modification in the mobile environment of network 108.

Network applications typically use application-level interfaces such as Windows sockets. A single call to an application-level API may generate several outgoing or incoming data packets at the transport, or media access layer. In prior mobile networks, if one of these packets is lost, the state of the entire connection may become ambiguous and the session must be dropped. In the preferred embodiment of the present invention providing RPCs, the Mobility Management Server 102 and the Mobile End Systems 104 share sufficient knowledge of the connection state to maintain a coherent logical link at all times -- even during physical interruption.

The Internet Mobility Protocol provided in accordance with the present invention compensates for differences between wire-line and other less reliable networks such as wireless. Adjusted frame sizes and protocol timing provide significant performance improvements over non-mobile-aware transports – dramatically reducing network traffic. This is important when bandwidth is limited or when battery life is a concern.

The Internet Mobility Protocol provided in accordance with the present invention also ensure the security of organization's data as it passes between the Mobile End System 104 and the Mobility Management Server 102 on the public wire-line networks or airway. The Internet Mobility Protocol provides a basic firewall function by allowing only authenticated devices access to the organizational network. The Internet Mobility Protocol can also certify and encrypt all communications between the mobility management system 102 and the Mobile End System 104.

The Remote Procedure Call protocol engine 240 on Mobile End System 104 of Figure 3 marshals TDI call parameters, formats the data, and sends the request to the Internet Mobility Protocol engine 244 for forwarding to Mobility Management Server 102 where the TDI Remote Procedure Call engine 240' executes the calls. Mobile End Systems 104 martial TDI call parameters according to the Remote Procedure Call protocol. When the Mobility Management Server 102 TDI Remote Procedure Call protocol engine 240' receives these RPCs, it executes the calls on behalf of the Mobile End System 104. The Mobility Management Server 102 TDI Remote Procedure Call protocol engine 240' shares the complete network state for each connected Mobile End System with the peer Mobile End System 104's RPC engine 240. In addition to performing remote procedure calls on behalf of the Mobile End Systems 104, the server RPC engine 240' is also responsible for system flow control, remote procedure call parsing, virtual address multiplexing (in coordination with services provided by address translator 220), remote procedure call transaction prioritization, scheduling, policy enforcement, and coalescing.

The Internet Mobility Protocol engine 244 performs reliable datagram services, sequencing, fragmentation, and re-assembly of messages. It can, when configured, also provide authentication, certification, data encryption and compression for enhanced privacy, security and throughput. Because the Internet Mobility Protocol engine 244 functions in power-sensitive environments using several different transports, it is power management aware and is transport independent.

Figure 3A shows an example process mobile interceptor 212 performs to communicate a TDI call to Mobility Management Server 102. Generally, the mobile interceptor RPC protocol engine 240 forwards marshaled TDI calls to the Internet Mobility Protocol engine 244 to be

transmitted to the Mobility Management Server 102. RPC protocol engine 240 does this by posting the RPC call to a queue maintained by the Internet Mobility Protocol engine 244 (block 302). To facilitate bandwidth management, the Internet Mobility Protocol engine 244 delays sending received RPC calls for some period of time ("the RPC coalesce time out period") (block 304). Typically, the RPC coalesce timeout is set between five and fifteen milliseconds as one example but is user configurable. This delay allows the RPC engine 240 to continue posting TDI calls to the Internet Mobility Protocol engine 244 queue so that more than one RPC call can be transmitted to the Mobility Management Server 102 in the same datagram (fragment).

When the coalesce timer expires, or the RPC protocol engine 240 determines that it will not be receiving more RPC calls (decision block 306), the RPC engine provides the Internet Mobility Protocol engine 244 with a request to flush the queue, coalesce the RPC calls into a single frame, and forward the frame to its peer (block 308). This coalescing reduces the number of transmissions -- enhancing protocol performance. However, the Internet Mobility Protocol may also decide to flush queue 244 based on other external criteria to further optimize performance. In the preferred embodiment, if a single RPC request will fill an entire frame, the IMP layer will automatically try to send the request to the peer.

As mentioned above, Mobility Management Server 102 proxy server also has an RPC protocol engine 212' and an Internet Mobility Protocol engine 244'. Figure 3B shows an example process performed by Mobility Management Server 102 upon receipt of an Internet Mobility Protocol message frame from Mobile End System 104. Once the frame is received by the Mobility Management Server 102, the Internet Mobility Protocol engine 244' reconstructs the frame if fragmented (due to the maximum

transmission size of the underlying transport) and then demultiplexes the contents of the message to determine which Mobile End System 104 it was received from. This demultiplexing allows the Internet Mobility Protocol 244' to provide the Remote Procedure Call engine 240' with the correct association- specific context information.

The Internet Mobility Protocol engine 244' then formulates the received message into a RPC receive indication system work request 354, and provides the Mobility Management Server 102 RPC engine 240' with the formulated work request and association-specific context information. When RPC protocol engine 240' receives work request 352, it places it into an association-specific work queue 356, and schedules the association to run by providing a scheduled request to a global queue 358. The main work thread of RPC engine 240' is then signaled that work is available. Once the main thread is awake, it polls the global queue 358 to find the previously queued association scheduled event. It then de-queues the event and beings to process the association-specific work queue 356.

On the association specific work queue 356 it finds the previously queued RPC receive indication work request The main thread then de-queues the RPC receive indication work request 356 and parses the request. Because of the coalescing described in connection with Figure 3A, the Mobility Management Server 102 often receives several RPC transactions bundled in each datagram. It then demultiplexes each RPC transaction back into distinct remote procedure calls and executes the requested function on behalf of Mobile End System 104. For performance purposes RPC engine 240' may provide a look ahead mechanism during the parsing process of the RPC messages to see if it can execute some of the requested transactions concurrently (pipelining).

How RPC Protocol Engine 240' Runs RPC Associations

Figure 4 is a flowchart of an example process for running RPC associations placed on an association work queue 356. When an RPC association is scheduled to run, the main thread for the RPC protocol engine 240' (which may be implemented as a state machine) de-queues the work request from global work queue 358 and determines the type of work request.

There are six basic types of RPC work requests in the preferred embodiment:

- schedule request;
- connect indication
- disconnect indication
- local terminate association
- "resources available" request; and
- ping inactivity timeout.

RPC protocol engine 240' handles these various types of requests differently depending upon its type. RPC protocol engine 240' tests the request type (indicated by information associated with the request as stored on global queue 358) in order to determine how to process the request.

If the type of work request is a "schedule request" (decision block 360), the RPC engine 240' determines which association is being scheduled (block 362). RPC engine 240' can determine this information from what is stored on global queue 358. Once the association is known, RPC engine 240' can identify the particular one of association work queues 356(1) ... 356(n) the corresponding request is stored on. RPC engine 240 retrieves the corresponding association control block (block 362), and calls a Process

Association Work task 364 to begin processing the work in a specific association's work queue 356 as previously noted.

Figure 5 shows example steps performed by the "process association work" task 364 of Figure 4. Once the specific association has been determined, this "process association work" task 364 is called to process the work that resides in the corresponding association work queue 356. If the de-queued work request (block 390) is an RPC receive request (decision block 392), it is sent to the RPC parser to be processed (block 394). Otherwise, if the de-queued work request is a pending receive request (decision block 396), the RPC engine 240' requests TDI 204' to start receiving data on behalf of the application's connection (block 398). If the de-queued work request is a pending connect request (decision block 400), RPC engine 240' requests TDI 204' to issue an application specified TCP (or other transport protocol) connect request (block 402). It then waits for a response from the TDI layer 204'. Once the request is completed by TDI 204', its status is determined and then reported back to the original requesting entity. As a performance measure, RPC engine 240' may decide to retry the connect request process some number of times by placing the request back on the associations-specific work queue (356) before actually reporting an error back to the requesting peer. This again is done in an effort to reduce network bandwidth and processing consumption.

The above process continues to loop until a "scheduling weight complete" test (block 404) is satisfied. In this example, a scheduling weight is used to decide how many work requests will be de-queued and processed for this particular association. This scheduling weight is a configuration parameter set by configuration manager 228, and is acquired when the association connect indication occurs (Figure 4, block 372). This value is configurable based on user or the physical identification of the machine.

Once the RPC engine is finished with the association work queue 356 (for the time at least), it may proceed to process dispatch queues (block 406) (to be discussed in more detail below). If, after processing work on the association's work queue 356, more work remains in the association work queue, the RPC engine 240' will reschedule the association to run again at a later time by posting a new schedule request to the global work queue 358 (Figure 4, decision block 366, block 368; Figure 5, decision block 408, block 410).

Referring once again to Figure 4, if the RPC work request is a "connect indication" (decision block 370), RPC engine 240' is being requested to instantiate a new association with a mobile peer (usually, but not always, the Mobile End System 104). As one example, the connect indication may provide the RPC engine 240' with the following information about the peer machine which is initiating the connection:

- physical identifier of the machine,
- name of the user logged into the machine,
- address of the peer machine, and
- optional connection data from the peer RPC engine 240.

In response to the connect indication (decision block 370), the RPC engine 240 calls the configuration manager 228 with these parameters. Configuration manager 228 uses these parameters to determine the exact configuration for the new connection. The configuration (e.g., association scheduling weight and the list of all applications that require non-default scheduling priorities along with those priorities) is then returned to the RPC engine 240' for storage and execution. RPC engine 240' then starts the new association, and creates a new association control block (block 372). As shown in Figure 5A the following actions may be taken:

- allocate an association control block (block 372A);

- initialize system wide resources with defaults (block 372B);
- get configuration overrides with current configuration settings (block 372C);
- initialize flags (block 372D);
- initialize the association-specific work queue (block 372E);
- initialize association object hash table (block 372F);
- initialize the coalesce timer (block 372G); and
- insert association control block into session table (block 372H).

A "disconnect indication" is issued by the Internet Mobility Protocol engine 244' to the RPC engine 240' when the Internet Mobility Protocol engine has determined that the association must be terminated. The RPC engine 240' tests for this disconnect indication (block 374), and in response, stops the association and destroys the association control block (block 376). As shown in Figure 5B, the following steps may be performed:

- mark the association as deleted to prevent any further processing of work that may be outstanding (block 376A);
- close all associated association objects including process, connection and address objects (block 376B);
- free all elements on work queue (block 376C);
- stop coalesce timer if running (block 376D);
- decrement association control block reference count (block 376E); and
- if the reference count is zero (tested for by block 376F):
- destroy association specific work queue,
- destroy object hash table,
- destroy coalesce timer,
- remove association control block from association table, and

- free control block (376G).

A "terminate session" request is issued when system 102 has determined that the association must be terminated. This request is issued by the system administrator, the operating system or an application. RPC engine 240' handles a terminate session request in the same way it handles a disconnect request (decision block 378, block 376).

In the preferred embodiment, the interface between the RPC engine 240' and the Internet Mobility Protocol engine 244' specifies a flow control mechanism based on credits. Each time one thread posts a work request to another thread, the called thread returns the number of credits left in the work queue. When a queue becomes full, the credit count goes to zero. By convention, the calling thread is to stop posting further work once the credit count goes to zero. Therefore, it is necessary to have a mechanism to tell the calling thread that "resources are available" once the queued work is processed and more room is available by some user configurable/pre-determined low-water mark in the queue. This is the purpose of the "resources available" work indication (tested for by decision block 380). As shown in Figure 5C, the following steps may be performed when the credit count goes to zero:

- mark association as "low mark pending" by setting the RPC_LMPQ_SEND_FLAG (block 379A). Once in this state:
- all received datagrams are discarded (block 379B);
- all received stream events are throttled by refusing to accept the data (block 379C) (this causes the TCP or other transport receive window to eventually close, and provides flow control between the Fixed End System 110 and the Mobility Management Server 102; before returning, the preferred embodiment jams a "pending receive request" to the front of the association specific work

queue 356 so that outstanding stream receive event processing will continue immediately once resources are made available).

- all received connect events are refused for passive connections (block 379D).

When the "resources available" indication is received by the RPC engine 240' (Figure 4, decision block 380), the RPC engine determine whether the association has work pending in its associated association work queue 356; if it does, the RPC engine marks the queue as eligible to run by posting the association to the global work queue 358 (block 382). If a pending receive request has been posted during the time the association was in the low mark pending state, it is processed at this time (in the preferred embodiment, the RPC engine 240 'continues to accept any received connect requests during this processing).

Referring once again to Figure 4, if RPC engine 240' determines that the Mobility Management Server 102 channel used for "ping" has been inactive for a specified period of time (decision block 384), the channel is closed and the resources are freed back to the system to be used by other processes (block 386).

RPC Parsing and Priority Queuing

Referring back to Figure 5, it was noted above that RPC engine parsed an RPC receive request upon receipt (see blocks 392, block 394). Parsing is necessary in the preferred embodiment because a single received datagram can contain multiple RPC calls, and because RPC calls can span multiple Internet Mobility Protocol datagram fragments. An example format for an RPC receive work request 500 is shown in Figure 6. Each RPC receive work request has at least a main fragment 502(1), and may have any number of additional fragments 502(2) .... 502(N). Main fragment

502(1) contains the work request structure header 503 and a receive overlay 504. The receive overlay 504 is a structure overlay placed on top of the fragment 502(1) by the Internet Mobility Protocol engine 244. Within this overlay 504 is a structure member called pUserData that points to the first RPC call 506(1) within the work request 500.

The Figure 6 example illustrates a work request 500 that contains several RPC calls 506(1), 506(2)...506(8). As shown in the Figure 6 example, an RPC work request 500 need not be contained in a contiguous block of memory or in a single fragment 502. In the example shown, a second fragment 502(2) and a third fragment 502(3) that are chained together to the main fragment 502(1) in a linked list.

Thus, RPC parser 394 in this example handles the following boundary conditions:

- each RPC receive request 500 may contain one or more RPC calls;
- one or more RPC calls 506 may exist in a single fragment 502;
- each RPC call 506 may exist completely contained in a fragment 502; and
- each RPC call 506 may span more than one fragment 502.

Figure 7 shows an example RPC parser process 394 to parse an RPC receive work request 500. In this example, the RPC parser 394 gets the first fragment 502(1) in the work request, gets the first RPC call 506(1) in the fragment, and parses that RPC call. Parser 394 proceeds through the RPC receive work request 500 and processes each RPC call 506 in turn. If the number of fragment bytes remaining in the RPC receive work request 500 fragment 502(1) is greater than the size of the RPC header 503, parser 394 determines whether the RPC call is fully contained within the RPC fragment 502 and thus may be processed (this may be determined by testing whether

the RPC call length is greater than the number of fragment bytes remaining). If the RPC call type is a chain exception, then the RPC call will handle the updating of the RPC parser 394 state. In the proxy server 224, the only RPC calls using the chain exception are the "datagram send" and "stream send" calls. This chain exception procedure is done to allow the RPC engine to avoid fragment copies by chaining memory descriptor lists together for the purpose of RPC send calls.

Once the parser 394 identifies an RPC call type, a pointer to the beginning of the RPC information is passed to the RPC engine 240 for execution. The RPC engine divides all TDI procedure calls into different priorities for execution. The highest priority calls are immediately executed by passing them to an RPC dispatcher 395 for immediate execution. All lower priority calls are dispatched to dispatch queues 510 for future processing. Each dispatch queue 510 represents a discrete priority.

In the preferred embodiment, mobile applications call the "open address" object and "open connection" object functions before executing other TDI networking functions. Therefore, the system assigns application level priorities during the "open address" object and "open connection" object calls. In the example embodiment, once an address or connection object is assigned a priority, all calls that are associated with that object are executed within that assigned priority.

If, for example, the RPC call is a TDI Open Address Object request or a TDI Open Connection Object Request, it is sent to the RPC dispatcher 395 for immediate execution. The Open Address and Open Connection object RPC calls provide access to a process ID or process name that are used to match against the information provided by the configuration manager 228 during the configuration requests that occurs within the

association connect indication described earlier. This is used to acquire configuration for the address or connection object.

In the preferred embodiment, all RPC calls have at least an address object or connection object as a parameter. When the call is made, the priority assigned to that specific object is used as the priority for the RPC call. The configuration assigned to the address or connection object determines which priority all associated RPC calls will be executed in. For example, if the assigned priority is "high," all RPC calls will be executed immediately without being dispatched to a dispatch queue 510. If the assigned priority is "1," all RPC calls will be placed into dispatch queue 510(1).

Referring once again to Figure 5, once the "process association work" task 364 process has completed executing its scheduled amount of association work (decision block 404), it checks to see if the dispatch queues require servicing (block 406). Figure 8 is a flowchart of example steps performed by the "process dispatch queues" block 406 of Figure 5 to process the dispatch queues 510 shown in Figure 7.

In this example, dispatch queues 510 are processed beginning with the highest priority queue (510(1) in this example) (block 408). Each queue 510 is assigned a weight factor. The weight factor is a configuration parameter that is returned by the configuration manager 228 when a Mobile End System 104 to Mobility Management Server 102 association is created. As one example, low priority dispatch queues 510 can have a weight factor of 4, and medium priority queues can have a weight factor of 8. High priority RPC calls do not, in this example, use weight factors because they are executed immediately as they are parsed.

RPC engine 240' loops through the de-queuing of RPC calls from the current queue until either the queue is empty or the queue weight

number of RPC calls has been processed (blocks 412-416). For each de-queued RPC call, the RPC dispatcher 395 is called to execute the call. The RPC dispatcher 395 executes the procedural call on behalf of the Mobile End System 104, and formulates the Mobile End System response for those RPC calls that require responses.

If, after exiting the loop, the queue still has work remaining (decision block 418), the queue will be marked as eligible to run again (block 420). By exiting the loop, the system yields the processor to the next lower priority queue (blocks 424, 410). This ensures that all priority levels are given an opportunity to run no matter how much work exists in any particular queue. The system gets the next queue to service, and iterates the process until all queues have been processed. At the end of processing all queues, the system tests to see if any queues have been marked as eligible to run – and if so, the association is scheduled to run again by posting a schedule request to the global work queue. The association is scheduled to run again in the "process global work" routine shown in Figure 4 above. This approach yields the processor to allow other associations that have work to process an opportunity run. By assigning each queue a weight factor, the system may be tuned to allow different priority levels unequal access to the Mobility Management Server 102's CPU. Thus, higher priority queues are not only executed first, but may also be tuned to allow greater access to the CPU.

Mobility Management Server RPC Responses

The discussion above explains how remote procedure calls are sent from the Mobile End System 104 to the Mobility Management Server 102 for execution. In addition to this type of RPC call, the Mobility Management Server 102 RPC engine 240' also supports RPC events and

RPC receive responses. These are RPC messages that are generated asynchronously as a result of association specific connection peer activity (usually the Fixed End System 110). Mobility Management Server 102 RPC engine 240' completes RPC transactions that are executed by the RPC dispatcher 395. Not all RPC calls require a response on successful completion. Those RPC calls that do require responses on successful completion cause the RPC dispatcher 395 to build the appropriate response and post the response to the Internet Mobile Protocol engine 244' to be returned to the peer Mobile End System 104. All RPC calls generate a response when the RPC call fails (the RPC receive response is the exception to above).

RPC events originate as a result of network 108 activity by the association specific connection (usually the Fixed End System 110). These RPC event messages are, in the preferred embodiment, proxied by the Mobility Management Server 102 and forwarded to the Mobile End System 104. The preferred embodiment Mobility Management Server 102 supports the following RPC event calls:

- Disconnect Event (this occurs when association-specific connected peer (usually the Fixed End System 110) issues a transport level disconnect request; the disconnect is received by the proxy server 224 on behalf of the Mobile End System 104, and the proxy server then transmits a disconnect event to the Mobile End System);
- Stream Receive Event (this event occurs when the association-specific connected peer (usually the Fixed End System 110) has sent stream data to the Mobile End System 104; the proxy server 224 receives this data on behalf of the Mobile End System 104,

and sends the data to the Mobile End System in the form of a Receive Response);

- Receive Datagram Event (this event occurs when any association-specific portal receives datagrams from a network peer (usually the Fixed End System 110) destined for the Mobile End System 104 through the Mobility Management Server 102; the proxy server 224 accepts these datagrams on behalf of the Mobile End System, and forwards them to the Mobile End System in the form of receive datagram events; and
- Connect Event (this event occurs when the association-specific listening portal receives a transport layer connect request (usually from the Fixed End System 110) when it wishes to establish a transport layer end-to-end connection with a Mobile End System 104; the proxy server 224 accepts the connect request on behalf of the Mobile End System, and then builds a connect event RPC call and forwards it to the Mobile End System).

Figure 9 shows how the RPC engine 240' handles proxy server-generated RPC calls. For high priority address and connection objects, the RPC engine 240' dispatches a send request to the Internet Mobility Protocol engine 244' immediately. The send request results in forwarding the RPC message to the peer Mobile End System 104. For lower priority objects, the Internet Mobility Protocol engine 244 send request is posted to an appropriate priority queue 510'. If the association is not scheduled to run, a schedule request is also posted to the global queue 358'. The Internet Mobility Protocol send request is finally executed when the dispatch queues are processed as described earlier in connection with Figures 5 & 8.

Example Internet Mobility Protocol

Internet Mobility Protocol provided in accordance with the present invention is a message oriented connection based protocol. It provides guaranteed delivery, (re)order detection, and loss recovery. Further, unlike other conventional connection oriented protocols (i.e. TCP), it allows for multiple distinct streams of data to be combined over a single channel; and allows for guaranteed, unreliable, as well as new message oriented reliable data to traverse the network through the single virtual channel simultaneously. This new message oriented level of service can alert the requester when the Internet Mobility Protocol peer has acknowledged a given program data unit.

The Internet Mobility Protocol provided in accordance with the present invention is designed to be an overlay on existing network topologies and technologies. Due to its indifference to the underlying network architecture, it is transport agnostic. As long as there is a way for packetized data to traverse between two peers, Internet Mobility Protocol can be deployed. Each node's network point of presence (POP) or network infrastructure can also be changed without affecting the flow of data except where physical boundary, policy or limitations of bandwidth apply.

With the help of the layer above, Internet Mobility Protocol coalesces data from many sources and shuttles the data between the peers using underlying datagram facilities. As each discrete unit of data is presented from the upper layer, Internet Mobility Protocol combines into a single stream and subsequently submits it for transmission. The data units are then forwarded to the peer over the existing network where upon reception, with the help from the layer above, the stream is demultiplexed back into multiple distinct data units. This allows for optimum use of available bandwidth, by generating the maximum sized network frames

possible for each new transmission. This also has the added benefit of training the channel once for maximum bandwidth utilization and have its parameters applied to all session level connections.

In rare instances where one channel is insufficient, the Internet Mobility Protocol further allows multiple channels to be established between the peers -- thus allowing for data prioritization and possibly providing a guaranteed quality of service (if the underlying network provides the service).

The Internet Mobility Protocol also provides for dynamically selectable guaranteed or unreliable levels of service. For example, each protocol data unit that is submitted for transmission can be queued with either a validity time period or a number of retransmit attempts or both. Internet Mobility Protocol will expire a data unit when either threshold is reached, and remove it from subsequent transmission attempts.

Internet Mobility Protocol's additional protocol overhead is kept minimal by use of variable length header. The frame type and any optional fields determine the size of the header. These optional fields are added in a specific order to enable easy parsing by the receiving side and bits in the header flag field denote their presence. All other control and configuration information necessary for the peers to communicate can be passed through the in-band control channel. Any control information that needs to be sent is added to the frame prior to any application level protocol data unit. The receiving side processes the control information and then passes the rest of the payload to the upper layer.

Designed to run over relatively unreliable network links where the error probability is relatively high, Internet Mobility Protocol utilizes a number of techniques to insure data integrity and obtain optimum network performance. To insure data integrity, a Fletcher checksum algorithm is

used to detect errant frames. This algorithm was selected due to the fact of its efficiency as well as its detection capability. It can determine not only bit errors, but also bit reordering. However, other alternate checksum algorithms maybe used in its place.

Sequence numbers are used to insure ordered delivery of data. Internet Mobility Protocol sequence numbers do not, however, represent each byte of data as in TCP. They represent a frame of data that can be, in one example implementation, as large as 65535 bytes (including the Internet Mobility Protocol header). They are 32 bits or other convenient length in one example to insure that wrap-around does not occur over high bandwidth links in a limited amount of time.

Combining this capability along with the expiration of data, retransmitted (retried) frames may contain less information than the previous version that was generated by the transmitting side. A frame id is provided to enable detection of the latest versioned frame. However, since data is never added in the preferred embodiment and each element removed is an entire protocol data unit, this is not a necessity for sequence assurance. In one example, the Internet Mobility Protocol will only process the first instance of a specific frame it receives -- no matter how many other versions of that frame are transmitted. Each frame created that carries new user payload is assigned its own unique sequence number.

Performance is gained by using of a sliding window technique -- thus allowing for more then one frame to be outstanding (transmitted) at a time before requiring the peer to acknowledge reception of the data. To insure timely delivery of the data, a positive acknowledgement and timer based retransmit scheme is used. To further optimize the use of the channel, a selective acknowledgement mechanism is employed that allows for fast retransmission of missing frames and quick recovery during lossy or

congested periods of network connectivity. In one example, this selective acknowledgement mechanism is represented by an optional bit field that is included in the header.

A congestion avoidance algorithm is also included to allow the protocol to back off from rapid retransmission of frames. For example, a round trip time can be calculated for each frame that has successfully transfer between the peers without a retransmit. This time value is averaged and then used as the basis for the retransmission timeout value. As each frame is sent, a timeout is established for that frame. If an acknowledgement for that frame is not received, and the frame has actually been transmitted, the frame is resent. The timeout value is then increased and then used as the basis for the next retransmission time. This retransmit time-out is bounded on both the upper and lower side to insure that the value is within a reasonable range.

Internet Mobility Protocol also considers the send and receive paths separately. This is especially useful on channels that are asymmetric in nature. Base on hysteresis, the Internet Mobility Protocol automatically adjusts parameters such as frame size (fragmentation threshold), number of frames outstanding, retransmit time, and delayed acknowledgement time to reduce the amount of duplicate data sent through the network.

Due to the fact that Internet Mobility Protocol allows a node to migrate to different points of attachment on diverse networks, characteristics (e.g., frame size) of the underlying network may change midstream. An artifact of this migration is that frames that have been queued for transmission on one network may no longer fit over the new medium the mobile device is currently attached to. Combining this issue with the fact that fragmentation may not be supported by all network infrastructures, fragmentation is dealt with at the Internet Mobility Protocol

level. Before each frame is submitted for transmission, Internet Mobility Protocol assesses whether or not it exceeds the current fragmentation threshold. Note that this value may be less than the current maximum transmission unit for performance reason (smaller frames have a greater likelihood of reaching its ultimate destination then larger frames). The tradeoff between greater protocol overhead versus more retransmissions is weighed by Internet Mobility Protocol, and the frame size may be reduced in an attempt to reduce overall retransmissions). If a given frame will fit, it is sent in its entirety. If not, the frame is split into maximum allowable size for the given connection. If the frame is retransmitted, it is reassessed, and will be refragmented if the maximum transmission unit has been reduced (or alternatively, if the maximum transmission unit actually grew, the frame may be resent as a single frame without fragmentation).

The protocol itself is orthogonal in its design as either side may establish or terminate a connection to its peer. In a particular implementation, however, there may be a few minor operational differences in the protocol engine depending on where it is running. For example, based on where the protocol engine is running, certain inactivity detection and connection lifetime timeouts may be only invoked on one side. To allow administrative control, Internet Mobility Protocol engine running on the Mobility Management Server 102 keeps track of inactivity periods. If the specified period of time expires without any activity from the Mobile End System 104, the Mobility Management Server 102 may terminate a session. Also, an administrator may want to limit the overall time a particular connection may be established for, or when to deny access base on time of day. Again these policy timers may, in one example implementation, be invoked only on the Mobility Management Server 102 side.

In one example implementation, the software providing the Internet Mobility Protocol is compiled and executable under Windows NT, 9x, and CE environments with no platform specific modification. To accomplish this, Internet Mobility Protocol employs the services of a network abstraction layer (NAL) to send and receive Internet Mobility Protocol frames. Other standard utility functions such as memory management, queue and list management, event logging, alert system, power management, security, etc are also used. A few runtime parameters are modified depending on whether the engine is part of a Mobile End System 104 or Mobility Management Server 102 system. Some examples of this are:

- Certain timeouts are only invoked on the Mobility Management Server 102
- Direction of frames are indicated within each frame header for echo detection
- Inbound connections may be denied if Mobile End System 104 is so configured
- Alerts only signaled on Mobility Management Server 102
- Power management enabled on Mobile End System 104 but is not necessary on the Mobility Management Server 102

The Internet Mobility Protocol interface may have only a small number of "C" callable platform independent published API functions, and requires one O/S specific function to schedule its work (other then the aforementioned standard utility functions). Communications with local clients is achieved through the use of defined work objects (work requests). Efficient notification of the completion of each work element is accomplished by signaling the requesting entity through the optional completion callback routine specified as part of the work object.

The Internet Mobility Protocol engine itself is queue based. Work elements passed from local clients are placed on a global work queue in FIFO order. This is accomplished by local clients calling a published Internet Mobility protocol function such as "ProtocolRequestwork()". A scheduling function inside of Internet Mobility Protocol then removes the work and dispatches it to the appropriate function. Combining the queuing and scheduling mechanisms conceal the differences between operating system architectures -- allowing the protocol engine to be run under a threaded based scheme (e.g., Windows NT) or in a synchronous fashion (e.g., Microsoft Windows 9x & Windows CE). A priority scheme can be overlaid on top of its queuing, thus enabling a guaranteed quality of service to be provided (if the underlying network supports it).

From the network perspective, the Internet Mobility Protocol uses scatter-gather techniques to reduce copying or movement of data. Each transmission is sent to the NAL as a list of fragments, and is coalesced by the network layer transport. If the transport protocol itself supports scatter-gather, the fragment list is passed through the transport and assembled by the media access layer driver or hardware. Furthermore, this technique is extensible in that it allows the insertion or deletion of any protocol wrapper at any level of the protocol stack. Reception of a frame is signaled by the NAL layer by calling back Internet Mobility Protocol at a specified entry point that is designated during the NAL registration process.

Example Internet Mobility Protocol Engine Entry Points

Internet Mobility Protocol in the example embodiment exposes four common entry points that control its startup and shutdown behavior. These procedures are:

1. Internet Mobility ProtocolCreate()

2. Internet Mobility ProtocolRun()

3. Internet Mobility ProtocolHalt()

4. Internet Mobility ProtocolUnload()

Example Internet Mobility ProtocolCreate()

The Internet Mobility ProtocolCreate() function is called by the boot subsystem to initialize the Internet Mobility Protocol. During this first phase, all resource necessary to start processing work must be acquired and initialized. At the completion of this phase, the engine must be in a state ready to accept work from other layers of the system. At this point, Internet Mobility Protocol initializes a global configuration table. To do this, it employs the services of the Configuration Manager 228 to populate the table.

Next it registers its suspend and resume notification functions with the APM handler. In one example, these functions are only invoked on the Mobile End System 104 side -- but in another implementation it might be desirable to allow Mobility Management Server 102 to suspend during operations. Other working storage is then allocated from the memory pool, such as the global work queue, and the global NAL portal list.

To limit the maximum amount of runtime memory required as well as insuring Internet Mobility Protocol handles are unique, Internet Mobility Protocol utilizes a 2-tier array scheme for generating handles. The globalConnectionArray table is sized based on the maximum number of simultaneous connection the system is configured for, and allocated at this time. Once all global storage is allocated and initialized, the global Internet Mobility Protocol state is change to _STATE_INITIALIZE_.

Example Internet Mobility ProtocolRun()

The Internet Mobility ProtocolRun() function is called after all subsystems have been initialized, and to alert the Internet Mobility Protocol subsystem that it is okay to start processing any queued work. This is the normal state that the Internet Mobility Protocol engine is during general operations. A few second pass initialization steps are taken at this point before placing the engine into an operational state.

Internet Mobility Protocol allows for network communications to occur over any arbitrary interface(s). During the initialization step, the storage for the interface between Internet Mobility Protocol and NAL was allocated. Internet Mobility Protocol now walks through the global portal list to start all listeners at the NAL. In one example, this is comprised of a two step process:

- Internet Mobility Protocol requests the NAL layer to bind and open the portal based on configuration supplied during initialization time; and
- Internet Mobility Protocol then notifies the NAL layer that it is ready to start processing received frames by registering the Internet Mobility ProtocolRCVFROMCB call back.
- A local persistent identifier (PID) is then initialized.

The global Internet Mobility Protocol state is change to _STATE_RUN_.

Example Internet Mobility ProtocolHalt

The Internet Mobility ProtocolHalt() function is called to alert the engine that the system is shutting down. All resources acquired during its operation are to be release prior to returning from this function. All Internet Mobility Protocol sessions are abnormally terminated with the reason code

set to administrative. No further work is accepted from or posted to other layers once the engine has entered into _STATE_HALTED_ state.

Example Internet Mobility ProtocolUnload()

The Internet Mobility ProtocolUnload() function is the second phase of the shutdown process. This is a last chance for engine to release any allocated system resources still being held before returning. Once the engine has returned from this function, no further work will be executed as the system itself is terminating

Example Internet Mobility Protocol handles

In at least some examples, using just the address of the memory (which contains the Internet Mobility Protocol state information) as the token to describe an Internet Mobility Protocol connection may be insufficient. This is mainly due to possibility of one connection terminating and a new one starting in a short period of time. The probability that the memory allocator will reassign the same address for different connections is high – and this value would then denote both the old connection and a new connection. If the original peer did not hear the termination of the session (i.e. it was off, suspended, out of range, etc.), it could possibly send a frame on the old session to the new connection. This happens in TCP and will cause a reset to be generated to the new session if the peer's IP addresses are the same. To avoid this scenario, Internet Mobility Protocol uses manufactured handle. The handles are made up of indexes into two arrays and a nonce for uniqueness. The tables are laid out as follows.

Table 1: an array of pointers to an array of connection object

Table 2 : an array of connection objects that contains the real pointers to the Internet Mobility Protocol control blocks.

This technique minimizes the amount of memory being allocated at initialization time. Table 1 is sized and allocated at startup. On the Mobile End System 104 side this allows allocation of a small amount of memory (the memory allocation required for this Table 1 on the Mobility Management Server 102 side is somewhat larger since the server can have many connections).

Table 1 is then populated on demand. When a connection request is issued, Internet Mobility Protocol searches through Table 1 to find a valid pointer to Table 2. If no entries are found, then Internet Mobility Protocol will allocate a new Table 2 with a maximum of 256 connection objects -- and then stores the pointer to Table 2 into the appropriate slot in Table 1. The protocol engine then initializes Table 2, allocates a connection object from the newly created table, and returns the manufactured handle. If another session is requested, Internet Mobility Protocol will search Table 1 once again, find the valid pointer to Table 2, and allocate the next connection object for the session. This goes on until one of two situations exist:

- If all the connection objects are exhausted in Table 2, a new Table 2 will be allocated, initialized, and a pointer to it will be placed in the next available slot in Table 1; and
- If all connection objects have been released for a specific Table 2 instance and all elements are unused for a specified period of time, the storage for that instance of Table 2 is released back to the memory pool and the associated pointer in Table 1 is zeroed to indicate that that entry is now available for use when the next connection request is started (if and only if no other connection object are available in other instances of Table 2).

Two global counters are maintained to allow limiting the total number of connections allocated. One global counter counts the number of current active connections; and the other keeps track of the number of unallocated connection objects. The second counter is used to govern the total number of connection object that can be created to some arbitrary limit. When a new Table 2 is allocated, this counter is adjusted downward to account for the number of objects the newly allocated table represents. On the flip side, when Internet Mobility Protocol releases a Table 2 instance back to the memory pool, the counter is adjusted upward with the number of connection objects that are being released.

Example Work Flow

Work is requested by local clients through the Internet Mobility ProtocolRequestWork() function. Once the work is validated and placed on the global work queue, the Internet Mobility ProtocolWorkQueueEligible() function is invoked. If in a threaded environment, the Internet Mobility Protocol worker thread is signaled (marked eligible) and control is immediately returned to the calling entity. If in a synchronous environment, the global work queue is immediately run to process any work that was requested. Both methods end up executing the Internet Mobility ProtocolProcessWork() function. This is the main dispatching function for processing work.

Since only one thread at a time may be dispatching work from the global queue in the example embodiment, a global semaphore may be used to protect against reentrancy. Private Internet Mobility Protocol work can post work directly to the global work queue instead of using the Internet Mobility ProtocolRequestWork() function.

A special case exists for SEND type work objects. To insure that the semantics of Unreliable Datagrams is kept, each SEND type work object can be queued with an expiry time or with a retry count. Work will be aged based on the expiry time. If the specified timeout occurs, the work object is removed from the connection specific queue, and is completed with an error status. If the SEND object has already been coalesced into the data path, the protocol allows for the removal of any SEND object that has specified a retry count. Once the retry count has been exceeded, the object is removed from the list of elements that make up the specific frame, and then returned to the requestor with the appropriate error status.

Example Connection Startup

Internet Mobility Protocol includes a very efficient mechanism to establish connections between peers. Confirmation of a connection can be determined in as little as a three-frame exchange between peers. The initiator sends an IMP SYNC frame to alert its peer that it is requesting the establishment of a connection. The acceptor will either send an IMP ESTABLISH frame to confirm acceptance of the connection, or send an IMP ABORT frame to alert the peer that its connection request has been rejected. Reason and status codes are passed in the IMP ABORT frame to aid the user in deciphering the reason for the rejection. If the connection was accepted, an acknowledgement frame is sent (possibly including protocol data unit or control data) and is forwarded to the acceptor to acknowledge receipt of its establish frame.

To further minimize network traffic, the protocol allows user and control data to be included in the initial handshake mechanism used at connection startup. This ability can be used in an insecure environment or in environments where security is dealt with by a layer below, such that the

Internet Mobility Protocol can be tailored to avert the performance penalties due to double security authentication and encryption processing being done over the same data path.

Example Data transfer

Internet Mobility Protocol relies on signaling from the NAL to detect when a frame has been delivered to the network. It uses this metric to determine if the network link in question has been momentarily flow controlled, and will not submit the same frame for retransmission until the original request has been completed. Some network drivers however lie about the transmission of frames and indicate delivery prior to submitting them to the network. Through the use of semaphores, the Internet Mobility Protocol layer detects this behavior and only will send another datagram until the NAL returns from the original send request

Once a frame is received by Internet Mobility Protocol, the frame is quickly validated, then placed on an appropriate connection queue. If the frame does not contain enough information for Internet Mobility Protocol to discern its ultimate destination, the frame is placed on the Internet Mobility Protocol socket queue that the frame was received on, and then that socket queue is place on the global work queue for subsequence processing. This initial demultiplexing allows received work to be dispersed rapidly with limited processing overhead.

Example Acquiescing

To insure minimal use of network bandwidth during periods of retransmission and processing power on the Mobility Management Server 102, the protocol allows the Mobility Management Server 102 to "acquiesce" a connection. After a user configurable period of time, the

Mobility Management Server 102 will stop retransmitting frames for a particular connection if it receives no notification from the corresponding Mobile End System 104. At this point, the Mobility Management Server 102 assumes that the Mobile End System 104 is in some unreachable state (i.e. out of range, suspended, etc), and places the connection into a dormant state. Any further work destined for this particular connection is stored for future delivery. The connection will remain in this state until one of the following conditions are met:

- Mobility Management Server 102 receives a frame from the Mobile End System 104, thus returning the connection to its original state;
- a lifetime timeout has expired;
- an inactivity timeout has expired; or
- the connection is aborted by the system administrator.

In the case that the Mobility Management Server 102 receives a frame from the Mobile End System 104, the connection continues from the point it was interrupted. Any work that was queued for the specific connection will be forwarded, and the state will be resynchronized. In any of the other cases, the Mobile End System 104 will be apprised of the termination of the connection once it reconnects; and work that was queued for the Mobile End System 104 will be discarded.

Example Connect and Send Requests

Figures 10A-10C together are a flowchart of example connect and send request logic formed by Internet mobility engine 244. In response to receipt from a command from RPC engine 240, the Internet Mobility Protocol engine 244 determines whether the command is a "connect" request (decision block 602). If it is, engine 244 determines whether

connection resources can be allocated (decision block 603). If it is not possible to allocate sufficient connection resources ("no" exit to decision block 603), engine 244 declares an error (block 603a) and returns. Otherwise, engine 244 performs a state configuration process in preparation for handling the connect request (block 603b).

For connect and other requests, engine 244 queues the connect or send request and signals a global event before return to the calling application (block 604).

To dispatch a connect or send request from the Internet Mobility Protocol global request queue, engine 244 first determines whether any work is pending (decision block 605). If no work is pending ("no" exit to decision block 605), engine 244 waits for the application to queue work for the connection by going to Figure 10C, block 625 (block 605a). If there is work pending ("yes" exit to decision block 605), engine 244 determines whether the current state has been established (block 606). If the state establish has been achieved ("yes" exit to decision block 606), engine 244 can skip steps used to transition into establish state and jump to decision block 615 of Figure 10B (block 606a). Otherwise, engine 244 must perform a sequence of steps to enter establish state ("no" exit to decision block 606).

In order to enter establish state, engine 244 first determines whether the address of its peer is known (decision block 607). If not, engine 244 waits for the peer address while continuing to queue work and transitions to Figure 10C block 625 (block 607a). If the peer address is known ("yes" exit to decision block 607), engine 244 next tests whether the requisite security context has been acquired (decision block 608). If not, engine 244 must wait for the security context while continuing to queue work and transitioning to block 625 (block 608a). If security context has already been acquired ("yes" exit to decision block 608), engine 244 declares a "state

pending" state (block 608b), and then sends an Internet Mobility Protocol sync frame (block 609) and starts a retransmit timer (block 610). Engine 244 determines whether the corresponding established frame was received (block 611). If it was not ("no" exit to decision block 611), engine 244 tests whether the retransmit time has expired (decision block 612). If the decision block has not expired ("no" exit to decision block 612), engine 244 waits and may go to step 625 (block 613). Eventually, if the established frame is never received (as tested for by block 611) and a total retransmit time expires (decision block 614), the connection may be aborted (block 614a). If the established is eventually received ("yes" exit to decision block 611), engine 244 declares a "state established" state (block 611a).

Once state establish has been achieved, engine 244 tests whether the new connection has been authenticated (decision block 615). If it has not been, engine 244 may wait and transition to step 625 (block 616). If the connection has been authenticated ("yes" exit to decision block 615), engine 244 tests whether authentication succeeded (decision block 617). If it did not ("no" exit to decision block 617), the connection is aborted (block 614a). Otherwise, engine 244 tests whether the peer transmit window is full (decision block 618). If it is ("yes" exit to decision block 618), engine 244 waits for acknowledgment and goes to step 625 (decision block 619). If the window is not full ("no" exit to decision block 618), engine 244 creates an Internet Mobility Protocol data frame (block 620) and sends it (block 621). Engine 244 then determines if the retransmit timer has started (decision block 622). If no, engine 244 starts the retransmit timer (block 623). Engine 244 loops through blocks 618-623 until there is no more data to send (as tested for by decision block 624). Engine 244 then returns to a sleep mode waiting for more work and returns to the global dispatcher (block 625).

Example Termination

Figure 11 is a flowchart of example steps performed by Internet Mobility Protocol engine 244 to terminate a connection. In response to a "terminate connection" request (block 626), the engine queues the request to its global work queue and returns to the calling application (block 626a). The terminate request is eventually dispatched from the Internet Mobility Protocol process global work queue for execution (block 627). Engine 244 examines the terminate request and determines whether the terminate request should be immediate or graceful (decision block 628). If immediate ("abort" exit to decision block 628), engine 244 immediately aborts the connection (block 629). If graceful ("graceful" exit to decision block 628), engine 244 declares a "state close" state (block 628a), and sends an Internet Mobility Protocol "Mortis" frame (block 630) to indicate to the peer that the connection is to close. Engine 244 then declares a "Mortis" state (block 630a) and starts the retransmit timer (block 631). Engine 244 tests whether the response of "post mortem" frame has been received from the peer (decision block 632). If not ("no" exit to decision block 632), engine 244 determines whether a retransmit timer has yet expired (decision block 633). If the retransmit timer is not expired ("no" exit to decision block 633), engine 244 waits and proceeds to step 637 (block 634). If the retransmit timer has expired ("yes" exit to decision block 633), engine 244 determines whether the total retransmit time has expired (decision block 635). If the total time is not yet expired ("no" exit to decision block 635), control returns to block 630 to resent the Mortis frame. If the total retransmit time has expired ("yes" exit to decision block 635), engine 244 immediately aborts the connection (block 635a).

Once a "post mortem" responsive frame has been received from the peer ("yes" exit to decision block 632), engine 244 declares a "post mortem"

state (block 632a), releases connection resources (block 636), and returns to sleep waiting for more work (block 637).

Example Retransmission

Figure 12 is a flowchart of example "retransmit" events logic performed by Internet Mobility Protocol engine 244. In the event that the retransmit timer has expired (block 650), engine 244 determines whether any frames are outstanding (decision block 651). If no frames are outstanding ("no" exit to decision block 651), engine 244 dismisses the timer (block 652) and returns to sleep (block 660). If, on the other hand, frames are outstanding ("yes" exit to decision block 651), engine 244 determines whether the entire retransmit period has expired (decision block 653). If it has not ("no" exit to decision block 653), the process returns to sleep for the difference in time (block 654). If the entire retransmit time period has expired ("yes" exit to decision block 653), engine 244 determines whether a total retransmit period has expired (decision block 655). If it has ("yes" exit to decision block 655) and this event has occurred in the Mobility Management Server engine 244' (as opposed to the Mobile End System engine 244), a dormant state is declared (decision block 656, block 656a). Under these same conditions, the Internet Mobility Protocol engine 244 executing on the Mobile End System 104 will abort the connection (block 656b).

If the total retransmit period is not yet expired ("no" exit to decision block 655), engine 244 reprocesses the frame to remove any expired data (block 657) and then retransmits it (block 658) -- restarting the retransmit timer as it does so (block 659). The process then returns to sleep (block 660) to wait for the next event.

Example Internet Mobility Protocol expiration of a PDU

Figure 12 block 657 allows for the requesting upper layer interface to specify a timeout or retry count for expiration of any protocol data unit (i.e. a SEND work request) submitted for transmission to the associated peer. By use of this functionality, Internet Mobility Protocol engine 244 maintains the semantics of unreliable data and provides other capabilities such as unreliable data removal from retransmitted frames. Each PDU (protocol data unit) 506 submitted by the layer above can specify a validity timeout and/or retry count for each individual element that will eventually be coalesced by the Internet Mobility Protocol engine 244. The validity timeout and/or retry count (which can be user-specified for some applications) are used to determine which PDUs 506 should not be retransmitted but should instead be removed from a frame prior to retransmission by engine 244.

The validity period associated with a PDU 506 specifies the relative time period that the respective PDU should be considered for transmission. During submission, the Internet Mobility Protocol RequestWork function checks the expiry timeout value. If it is non-zero, an age timer is initialized. The requested data is then queued on the same queue as all other data being forwarded to the associated peer. If the given PDU 506 remains on the queue for longer than the time period specified by the validity period parameter, during the next event that the queue is processed, the given (all) PDU(s) that has an expired timeout is removed and completed locally with a status code of "timeout failure" rather than being retransmitted when the frame is next retransmitted. This algorithm ensures that unreliable data being queued for transmission to the peer will not grow stale and/or boundlessly consume system resources.

In the example shown in Figure 12A, at least three separate PDUs 506 are queued to Internet Mobility Protocol engine 244 for subsequent processing. PDU 506(1) is queued without an expiry time denoting no timeout for the given request. PDU 506(2) is specified with a validity period of 2 seconds and is chronologically queued after PDU 506(1). PDU 506(n) is queued 2.5 seconds after PDU 506(2) was queued. Since the act of queuing PDU 506(n) is the first event causing processing of the queue and PDU 506(2) expiry time has lapsed, PDU 506(2) is removed from the work queue, completed locally and then PDU 506(n), is placed on the list. If a validity period was specified for PDU 506(n) the previous sequence of events would be repeated. Any event (queuing, dequeuing, etc) that manipulates the work queue will cause stale PDUs to be removed and completed.

As described above, PDUs 506 are coalesced by the Internet Mobility Protocol Engine 244 transmit logic and formatted into a single data stream. Each discrete work element, if not previously expired by the validity timeout, is gathered to formulate Internet Mobility Protocol data frames. Internet Mobility Protocol Engine 244 ultimately sends these PDUs 506 to the peer, and then places the associated frame on a Frames-Outstanding list. If the peer does not acknowledge the respective frame in a predetermined amount of time (see Figure 12 showing the retransmission algorithm), the frame is retransmitted to recover from possibly a lost or corrupted packet exchange. Just prior to retransmission, the PDU list that the frame is comprised of is iterated through to determine if any requests were queued with a retry count. If the retry count is non zero, and the value is decremented to zero, the PDU 506 is removed from the list, and the frames header is adjusted to denote the deletion of data. In this fashion, stale data, unreliable data, or applications employing their own

retransmission policy are not burdened by engine 244's retransmission algorithm.

In the Figure 12B example, again three separate PDUs 506 are queued to Internet Mobility Protocol engine 244 for subsequent processing. PDU 506(1) is queued without a retry count. This denotes continuous retransmission attempts or guaranteed delivery level of service. PDU 506(2) is queued with a retry count of 1 and is chronologically queued after PDU 506(1). PDU 506(n) is queued sometime after PDU 506(2). At this point, some external event (e.g., upper layer coalesce timer, etc.) causes engine 244's send logic to generate a new frame by gathering enough PDUs 506 from the work queue to generate an Internet Mobility Protocol data frame 500. The frame header 503 is calculated and stamped with a retry ID of 0 to denote that this is the first transmission of the frame. The frame is then handed to the NAL layer for subsequent transmission to the network. At this point a retransmit timer is started since the frame in question contains a payload. For illustration purposes it is assumed that an acknowledgement is not received from the peer for a variety of possible reasons before the retransmit timer expires. The retransmit logic of engine 244 determines that the frame 500 in question is now eligible for retransmission to the network. Prior to resubmitting the frame to the NAL layer, engine 244's retransmit logic iterates through the associated list of PDUs 506. Each PDU's retry count is examined and if non-zero, the count is decremented. In the process of decrementing PDU 506(2)'s retry count, the retry count becomes zero. Because PDU 506(2)'s retry count has gone to zero, it is removed from the list and completed locally with a status of "retry failure." The frame header 503 size is then adjusted to denote the absence of the PDU 506(2)'s data. This process is repeated for all remaining PDUs. Once the entire frame 500 is reprocessed to produce an

"edited" frame 500', the retry ID in the header is incremented and the resultant datagram is then handed to the NAL layer for subsequent (re)transmission.

Example Reception

Figures 13A-13D are a flowchart of example steps performed by Internet Mobility Protocol engine 244 in response to receipt of a "receive" event. Such receive events are generated when an Internet Mobility Protocol frame has been received from network 108. In response to this receive event, engine 244 pre-validates the event (block 670) and tests whether it is a possible Internet Mobility Protocol frame (decision block 671). If engine 244 determines that the received frame is not a possible frame ("no" exit to decision block 671), it discards the frame (block 672). Otherwise ("yes" exit to decision block 671), engine 244 determines whether there is a connection associated with the received frame (decision block 673). If there is a connection associated with the received frame ("yes" exit to decision block 673), engine 244 places the work on the connection receive queue (block 674), marks the connection as eligible to receive (block 675), and places the connection on the global work queue (block 676). If no connection has yet been associated with the received frame ("no" exit to decision block 673), engine 244 places the received frame on the socket receive queue (block 677) and places the socket receive queue on the global work queue (block 678). In either case, engine 244 signals a global work event (block 679). Upon dispatching of a "receive eligible" event from the global work queue (see Figure 13B), engine 244 de-queues the frame from the respective receive queue (block 680). It is possible that more than one IMP frame is received and queued before the Internet Mobility Protocol engine 244 can start de-queuing the messages.

Engine 244 loops until all frames have been de-queue (blocks 681, 682). Once a frame has been de-queued ("yes" exit to decision block 681), engine 244 validates the received frame (block 683) and determines whether it is okay (decision block 684). If the received frame is invalid, engine 244 discards it (block 685) and de-queues the next frame from the receive queue (block 680). If the received frame is valid ("yes" exit to decision block 684), engine 244 determines whether it is associated with an existing connection (block 686). If it is not ("no" exit to decision block 686), engine 244 tests whether it is a sync frame (decision block 687). If it is not a sync frame ("no" exit to decision block 687), the frame is discarded (block 685). If, on the other hand, a sync frame has been received ("yes" exit to decision block 687), engine 244 processes it using a passive connection request discussed in association with Figures 14A and 14B (block 688).

If the frame is associated with a connection ("yes" exit to decision block 686), engine 244 determines whether the connection state is still active and not "post mortem" (decision block 689). If the connection is already "post mortem," the frame is discarded (block 685). Otherwise, engine 244 parses the frame (block 690) and determines whether it is an abort frame (decision block 691). If the frame is an abort frame, engine 244 immediately aborts the connection (block 691a). If the frame is not an abort frame ("yes" exit to decision block 691), engine 244 processes acknowledgment information and releases any outstanding send frames (block 692). Engine 244 then posts the frame to any security subsystem for possible decryption (block 693). Once the frame is returned from the security subsystem engine 244 processes any control data (block 694). Engine 244 then determines whether the frame contains application data (decision block 695). If it does, this data is queued to the application layer (block 696). Engine 244 also determines whether the connection's state is

dormant (block 697 and 697a -- this can happen on Mobility Management Server engine 244' in the preferred embodiment), and returns state back to established.

If the frame is possibly a "Mortis" frame ("yes" exit to decision block 698), engine 244 indicates a "disconnect" to the application layer (block 699) and enters the "Mortis" state (block 699a). It sends a "post mortem" frame to the peer (block 700), and enters the "post mortem" state (block 700a). Engine 244 then releases connection resources (block 701) and returns to sleep waiting for more work (block 702). If the parsed frame is a "post mortem" frame ("yes" exit to decision block 703), blocks 700a, 701, 702 are executed. Otherwise, control returns to block 680 to dequeue the next frame from the receive queue (block 704).

Example Passive Connections

Blocks 14A-14B are together a flowchart of example steps performed by Internet Mobility Protocol engine 244 in response to a "passive connection" request. Engine 244 first determines whether there is another connection for this particular device (block 720). If there is ("yes" exit to decision block 720), the engine determines whether it is the initial connection (decision block 721). If peer believes the new connection is the initial connection ("yes" exit to decision block 721), engine 244 aborts the previous connections (block 722). If not the initial connection ("no" exit to decision block 721), engine 244 tests whether the sequence and connection ID match (decision block 723). If they do not match ("no" exit to decision block 723), control returns to decision block 720. If the sequence and connection ID do match ("yes" exit to decision block 723), engine 244 discards duplicate frames (block 724) and returns to step 680 of Figure 13B (block 725).

If there is no other connection ("no" exit to decision block 720), engine 244 determines whether it can allocate connection resources for the connection (decision block 726). If it cannot, an error is declared ("no" exit to decision block 726, block 727), and the connection is aborted (block 728). If it is possible to allocate connection resources ("yes" exit to decision block 726), engine 244 declares a "configure" state (block 726a) and acquires the security context for the connection (block 730). If it was not possible to acquire sufficient security context ("no" exit to decision block 731), the connection is aborted (block 728). Otherwise, engine 244 sends an established frame (block 732) and declares the connection to be in state "establish" (block 732a). Engine 244 then starts a retransmitter (block 733) and waits for the authentication process to conclude (block 734). Eventually, engine 244 tests whether the device and user have both been authenticated (block 735). If either the device or the user is not authenticated, the connection is aborted (block 736). Otherwise, engine 244 indicates the connection to the listening application (block 737) and gets the configuration (block 738). If either of these steps do not succeed, the connection is aborted (decision block 739, block 740). Otherwise, the process returns to sleep waiting for more work (block 741).

Example Abnormal Termination

Figures 15A and 15B are a flowchart of example steps performed by the Internet Mobility Protocol engine 244 in response to an "abort" connection request. Upon receipt of such a request from another process (block 999) and dispatched via the queue (block 1000), engine 244 determines whether a connection is associated with the request (decision block 1001). If it is ("yes" exit to decision block 1001), engine 244 saves the original state (block 1002) and declares an "abort" state (block 1002a).

Engine 244 then determines whether the connection was indicated to the RPC engine (decision block 1003) -- and if so, indicates a disconnect event(block 1004). Engine 244 then declares a "post mortem" state (block 1003a), releases the resources previously allocated to the particular connection (block 1005), and tests whether the original state is greater than the state pending (decision block 1006). If not ("no" exit to decision block 1006), the process transitions to block 1012 to return to the calling routine (block 1007). Otherwise, engine 244 determines whether the request is associated with a received frame (decision block 1008). If the abort request is associated with a received frame, and the received frame is an abort frame (decision block 1009), the received frame is discarded (block 1010). Otherwise engine 244 will send an abort frame (block 1011) before returning to the calling routine (block 1012).

Example Roaming Control

Referring once again to Figure 1, mobile network 108 may comprise a number of different segments providing different network interconnects (107a-107k corresponding to different wireless transceivers 106a-106k). In accordance with another aspect of the present invention, network 108 including Mobility Management Server 102 is able to gracefully handle a "roaming" condition in which a Mobile End System 104 has moved from one network interconnect to another. Commonly, network 108 topographies are divided into segments (subnets) for management and other purposes. These different segments typically assign different network (transport) addresses to the various Mobile End Systems 104 within the given segment.

It is common to use a Dynamic Host Configuration Protocol (DHCP) to automatically configure network devices that are newly activated on such a subnet. For example, a DHCP server on the sub-net typically provides its

clients with (among other things) a valid network address to "lease". DHCP clients may not have permanently assigned, "hard coded" network addresses. Instead, at boot time, the DHCP client requests a network address from the DHCP server. The DHCP server has a pool of network addresses that are available for assignment. When a DHCP client requests an network address, the DHCP server assigns, or leases, an available address from that pool to the client. The assigned network address is then owned" by the client for a specified period ("lease duration"). When the lease expires, the network address is returned to the pool and becomes available for reassignment to another client. In addition to automatically assigning network addresses, DHCP also provides netmasks and other configuration information to clients running DHCP client software. More information concerning the standard DHCP protocol can be found in RFC2131.

Thus, when a Mobile End System 104 using DHCP roams from one subnet to another, it will appear with a new network address. In accordance with one aspect of the present invention, Mobile End Systems 104 and Mobility Management Server 102 take advantage of the automatic configuration functionality of DHCP, and coordinate together to ensure that the Mobility Management Server recognizes the Mobile End System's "new" network address and associates it with the previously-established connection the Mobility Management Server is proxying on its behalf.

One example embodiment uses standard DHCP Discover/Offer client-server broadcast messaging sequences as an echo request-response, along with other standard methodologies in order to determine if a Mobile End System 104 has roamed to a new subnet or is out of range. In accordance with the standard DHCP protocol, a Mobile End System 104 requiring a network address will periodically broadcast client identifier and

hardware address as part of a DHCP Discover message. The DHCP server will broadcast its Offer response (this message is broadcast rather than transmitted specifically to the requesting Mobile End System because the Mobile End System doesn't yet have a network address to send to). Thus, any Mobile End System 104 on the particular subnet will pick up any DHCP Offer server response to any other Mobile End System broadcast on the same subnet.

This example embodiment provides DHCP listeners to monitor the DHCP broadcast messages and thereby ascertain whether a particular Mobile End System 104 has roamed from one subnet to another and is being offered the ability to acquire a new network address by DHCP. Figure 16 shows example DHCP listener data structures. For example, a Mobile End System listener data structure 902 may comprise:

- a linked list of server data structures,
- an integer transaction ID number (xid),
- a counter ("ping"), and
- a timeout value.

A server data structure 904 may comprise a linked list of data blocks each defining a different DHCP server, each data block comprising:

- a pointer to next server,
- a server ID (network address of a DHCP server),
- an address (giaddr) of a BOOTP relay agent recently associated with this DHCP server,
- a "ping" value (socket -> ping), and
- a flag.

These data structures are continually updated based on DHCP broadcast traffic appearing on network 108. The following example functions can be used to maintain these data structures:

- roamCreate() [initialize variables]
- roamDeinitialize() [delete all listeners]
- roamStartIndications() [call a supplied callback routine when a Mobile End System has roamed or changed interfaces, to give a registrant roaming indications]
- roamStopIndications() [remove the appropriate callback from the list, to stop giving a registrant roaming indications]
- Interface Change [callback notification from operating system indicating an interface has changed its network address]
- Listener Signal [per-interface callback from a Listener indicating a roaming or out-of-range or back-in-range condition].

Additionally, a refresh process may be used to update Listeners after interface changes.

In the preferred embodiment, all Mobile End Systems 104 transmit the same Client Identifier and Hardware Address in DHCP Discover requests. This allows the listener data structures and associated processes to distinguish Mobile End System-originated Discover requests from Discover requests initiated by other network devices. Likewise, the DHCP server will broadcast its response, so any Mobile End System 104 and/or the Mobility Management Server 102 will be able to pick up the DHCP server Offer response to any other Mobile End System. Since multiple DHCP servers can respond to a single DHCP Discover message, the listener data structures shown in Figure 16 store each server response in a separate data block, tied to the main handle via linked list.

Upon receiving a Discover request having the predetermined Client Hardware Address and Client Identifier, the preferred embodiment recognizes this request as coming from a Mobile End System 104. If the message also has a BOOTP relay address set to zero, this indicates that the message originated on the same subnet as the listener. Listeners may ignore all DHCP Offers unless they have a transaction ID (xid) matching that of a Discover message recently sent by a Mobile End System 104. The listener can determine that a Mobile End System 104 has roamed if any response comes from a known server with a new BOOTP relay agent ID and/or offered network address masked with an offered subnet mask. Listeners add new servers to the Figure 16 data structures only after receiving a positive response from an old server. If a listener receives responses from new server(s) but none from an old server, this may indicate roaming (this can be a configurable option). If the listener fails to receive responses from new or old servers, the listener is out of range (this determination can be used to signal an upper layer such as an application to halt or reduce sending of data to avoid buffer overflow).

If the listener never receives a response from any server, there is no point of reference and thus impossible to determine whether roaming has occurred. This condition can be handled by signaling an error after a timeout and allowing the caller to retry the process. The preferred embodiment determines that a Mobile End System 104 has roamed if any response has come from a known server with a new BOOTP relay agent ID (or a new offered network address when masked with offered subnet mask). If the listener data structures see responses from new servers but none from an old server, it is possible that roaming has occurred, but there must be a delay before signaling, in order to wait for any potential responses from the old servers. If there are no responses from new or old servers, then the

Mobile End System 104 is probably out of range and Mobility Management Server 102 waits for it to come back into range.

Figure 17 is a flowchart of example steps of a Listener process of the preferred embodiment. Referring to Figure 17, a DHCP listener process is created by allocating appropriate memory for the handle, opening NAL sockets for the DHCP client and server UDP ports, and setting receive callbacks for both. A timer is then set (block 802) and then the process enters the "Wait" state to wait for a roaming related event (block 804). Three external inputs can trigger an event:

- a DHCP server packet is received;
- a DHCP client packet sent by another Mobile End System is received
- a timer timeout occurs.

If a DHCP server packet has been received, the packet is examined to determine whether its client identifier matches the predetermined client ID (decision block 806). If it does not, it is discarded. However, if the packet does contain the predetermined ID, a test is performed to determine whether the packet is a DHCP Offer packet (decision block 808). Offer packets are rejected unless they contain a transaction ID matching a recently sent DHCP Discover sequence.

If the packet transaction ID matches (block 810), then a test is made as to whether the server sending the DHCP offer packet is known (i.e., the server ID is in the listener data structure shown in Figure 16) (block 812). If the server ID is not on the list ("no" exit to decision block 812), it is added to the list and marked as "new" (or "first" if it is the first server on the list) (block 822). If the server is already on the list ("Y" exit to decision block 812), a further test is performed to determine whether the packet BOOTP relay address ("GIADDR") matches the server address

("GIADDR") (decision block 814). If there is no match, then the Offer packet must be originating from a different subnet, and it is determined that a "hard roam" has occurred (block 816). The caller application is signaled that there has been a roam. If, on the other hand, decision block 814 determines there is a match in BOOTP relay addresses, then no roam has occurred, the listener process stamps the server receive time, resets "new" flags for all other servers on the list, and stores the current ping number with the server (block 818, 820). The process then returns to "wait" period.

If the event is a received client packet, the listener process determines whether the packet has the predetermined client ID, is a DHCP Discover packet and has a BOOTP relay address (GIADDR) of 0 (blocks 824, 826, 828). These steps determine whether the received packet is DHCP Discover message sent by another Mobile End System 104 on the same sub-net as the listener. If so, the listener process then sets the transaction ID to the peer's transaction ID (block 830) for use in comparing with later-received DHCP Offer packets, calls a ping check (block 834) and resets the timer (block 836).

In response to a timer timeout, the process calls a "ping check" (block 838). "Pings" in the preferred embodiment are DHCP Discover packets with a random new xid. Example steps for this ping check 838 are shown in Figure 17A. The purpose of the ping check routine is to determine if a "soft roam" condition has occurred (i.e., a Mobile End System has temporarily lost and then regained contact with a sub-net, but has not roamed to a different sub-net). The process determines whether there is a sub-net roam condition, an out-of-range condition, or a "no server" condition. In other words:

- Has a Mobile End System roamed from one sub-net to another?
- Is a Mobile End System out of range?

- Is a DHCP server absent?

These conditions are determined by comparing Mobile End System prior "ping" response with the current "ping" response (decision blocks 846, 850). For example, if the current ping number minus the old server's last ping response is greater than the sub-net server pings and there is at least one server marked "new," there has been a sub-net roam to a different server. The result of this logic is to either signal a subset roam, and out of range condition or a no server condition (or none of these) to the calling process.

Figure 18 shows a flowchart of example steps performed by a Mobile End System 104 roaming control center. To enable roaming at the Mobile End System 104, the list of known addresses is initialized to zero (block 850) and an operating system interface change notification is enabled (block 852). The process then calls the operating system to get a list of current addresses that use DHCP (block 854). All known addresses no longer in the current list have their corresponding listeners closed (block 856). Similarly, the process opens listeners on all current but not known interfaces (block 858). The process then signals "roam" to registrants (block 860).

When the listener process of Figure 17 signals (block 862), the process determines whether the signal indicates a "roam", "out of range" or "back in range" condition (decision block 864, 870, 874). A roam signal ("yes" exit to decision block 864) causes the process to close corresponding listener 866 and call the operating system to release and renew DHCP lease to a network address (block 868). If the listener signals "out of range" (decision block 870), the process signals this condition to registrants (block 872). If the signal is a "back in range" (decision block 874), then this condition is signaled to all registrants (block 876). Upon receiving a

disabled roam command (block 878), the process closes all listeners (block 880) and disables the operating system interface change notification (block 882).

Example Interface Assisted Roaming Listener

A further, interface-based listener feature enables roaming across network points of attachment on the same network or across different network media. This interface-based listener feature operates without requiring the beaconing techniques described above, while permitting the system to fall back on beaconing if the underlying interface(s) is unable to support the appropriate signaling.

In this further embodiment, an interface-based listener integrates information from network interface adapters (e.g., via a low level interface roaming driver) with information available from network stacks to determine whether a mobile node has moved to a new Network Point of Attachment. Figures 19A & 19B show an example listener algorithm that may be used to efficiently determine the migration path of the mobile node. This process is shown using a single network interface connected to a single network medium, but can be used by itself or in conjunction with other roaming algorithms to traverse across many diverse network media and interfaces (e.g., to create a self-healing infrastructure using redundant paths).

Referring to Figure 19A, at system initialization time or when the network adapter driver loads (Figure 19A, block 2000), low-level interface roaming drivers register with the roaming control center module of Figure 18 (block 2010). Such registration (which is made via the function crRegisterCardHandler() in the example embodiment) provides entry points for:

- open,
- close,
- get status, and
- a Boolean set to TRUE if the driver can notify the registrant of changes in status, and FALSE if the roaming control center module should use timer-based (or other) polling to check status.

The example embodiment function crRegisterCardHandler() also provides a interface description string or token that can be used by the roaming control center module for preliminary match-ups to the correct roaming driver. A default roaming driver may also be installed for interfaces that use an O/S generic mechanism for signaling/querying media connectivity as well as changes to network point of attachments.

In the example embodiment, when an interface's state becomes enabled (i.e. access to the network is now possible) (block 2020), the roaming control center tries to enable Interface Assisted Roaming (IAR) according to the following steps (please note however, that the steps may be interchanged or either might be omitted based on the design of the operating system (O/S) and/or the hosting device being used in a particular application):

1. If a generic handler is installed, a call to the generic crOpenInstance() handler is made. The generic handler queries the low-level adapter driver to see if it can generically support signaling the status of media connectivity as well as any changes to the network point of attachment (block 2030). If the interface driver is unable to support this functionality generically ("no" exit to block 2030), an error status is returned to the caller to indicate that it should use an alternative mechanism for acquiring signaling information.

2. If the generic handler returns an error ("no" exit to block 2030), a search is made with the token of the activated interface through the currently registered roaming drivers (block 2040). If the interface matches one of the tokens that was registered during crRegisterCardHandler() phase (block 2050), the roaming control center calls the specific crOpenInstance() for that instance of the adapter. This function attempts to open the low level driver, poll once for status (media connectivity, and the network point of attachment ID), and set the periodic polling timer (if applicable). If the low-level driver does not support the requests for some reason, an error is returned indicating that the roaming control center should use an alternate mechanism for acquiring signaling information.

3. If either of the previous steps is unable to achieve the required functionality, an error is returned to the roaming control center to signal that it should not use the IAR functionality and fall back to other roaming algorithms, such as the beaconing listener shown in Figure 17 & 17A, Mobile IP, or in some cases the currently attached network itself deals with roaming ("no" exit to block 2050, block 2999). Otherwise Interface Assisted Roaming is enabled (block 2060) and the roaming control center follows the algorithm outlined below.

Initially, the interface-assisted listener records current media connectivity status and network point of attachment identification information in a local data store (block 2060). Assuming the interface assisted subsystem is successful in providing roaming feedback, the subsystem waits for a status event (block 2100). The event can comprise, for example:

- a callback from the low level roaming driver,
- a timed poll interval (blocks 2070, 2090), or

- a hint from network level activity (i.e. trouble transmitting/receiving) (block 2080).

If the status of the interface signifies either a change in medium connectivity has occurred, or a change in network point of attachment ("yes" exit to block 2110 or 2120 of Figure 19B), any clients of the roaming control center are notified of the state change using the following rules:

1. If the status signifies a change from being connected to the underlying network medium to being detached ("yes" exit to block 2120) and there are no other paths to the peer, the listener concludes that the mobile end system has lost its connection, and the roaming control center signals its clients with a status of ROAM_SIGNAL_OUT_OF_CONTACT (block 2140).

2. If the status signifies that the interface has been reconnected to the medium, and the network point of attachment has not changed ("no" exit to block 2150 after "no" exit to block 2120) and a ROAM_SIGNAL_OUT_OF_CONTACT was previously signaled, this indicates that the mobile end system had previously lost but has now reestablished contact with a particular network point of attachment. In this case, the roaming control center will revalidate any network address it may have registered or acquired for proper access (block 2170), and signals ROAM_SIGNAL_ROAM_SAME_SUBNET (block 2180) to alert the roaming control center clients that a reattachment has occurred and that they should take whatever steps necessary to quickly reestablish transport level communications. For example, during the disruption in service it is possible that some data may have been lost -- and the clients may need to act to recover such lost data.

3. If the status signifies that the interface is attached to the medium but the network point of attachment has changed ("yes" exit to block 2150),

the roaming control center will signal its clients that a roaming condition has occurred. To more efficiently support handoffs between network point of attachments, the roaming control center in this example employs the use of a learning algorithm along with a local data-store. The data-store is normally populated dynamically (i.e. learning), but it can be seeded with static information (i.e., already learned information) to improve performance. The data-store itself maintains a list of network points of attachment identifiers, along with information such as network and media access address, network mask, etc. This "network topology map" assists the roaming control center in deciding the correct signal to generate to its clients.

Determination of the correct signal is done in the following manner in the example embodiment:

a) A search is made through the network topology map data-store to determine if the interface has already visited this particular network point of attachment (block 2190). If a match is found ("yes" exit to block 2200), a further check is made to see if the network point of attachment is on the same network segment as the one that the interface was previously associated with. If the network segment is the same, the roaming control center generates a ROAM_SIGNAL_ROAM_SAME_SUBNET. This alerts the roaming control center clients that a handoff occurred and it should take whatever steps necessary to quickly reestablish transport level communications as during the handoff it is possible that some data may have been lost.

b) If during the search a match is found, but the new network point of attachment is not on the same network segment, then the listener concludes that the mobile end system has roamed to a different subnetwork. In this case, the roaming control center:

- acquires an address that is usable on the new network segment (block 2220). This may entail registering the current address to be valid on the new segment, (re)acquiring an address from a local server, having one statically defined, or using heuristics to determine that an address that was previously assigned is still valid. In the latter case, the roaming control center may determine that the interface is changing between a given set of network point of attachments and may not immediately relinquish or de-register the network address for performance reasons. In this example, there is a difference between acquiring an address on the network (e.g., via DHCP) or registering the address on the local network (e.g., via a foreign agent in Mobile IP). The roaming entity either (re)acquires (e.g., possibly establishing/updating a lease with the DHCP server) or registers the current address with a foreign agent (Mobile IP).
- Generates a ROAM_SIGNAL_ROAM signal to its clients (block 2230) indicating roaming to a different subnet.

c) If the search yields no match ("no" exit to block 2200), a new record is created in the local data-store populated with the network point of attachment's identifier, media access address, network mask and other ancillary information (block 2210). The roaming control center then executes blocks 2220 and 2230 to acquire and register a network address, and to generate a "roam" signal.

Since the interface-assisted roaming technique described above gives access to the underlying interface information, it is possible to employ an additional set of policy parameters (defined by the user and/or the system) that can enable automatic efficient selection of alternate valid network paths. If there is more than one network available at a time, the subsystem

can choose the path(s) with the least cost associated with it (i.e., a wide area network connection versus a local area connection). This can be done by a number of metrics such as, for example, bandwidth, cost (per byte), and/or quality of service. Such "least cost routing" techniques can provide advantages in terms of network connection quality, efficiency, and reduction in frame loss. For example, it is possible to provide a "make before break" handoff scheme based on other heuristics available (media connectivity, signal strength, retransmission rate, etc.), thus allowing continuous packet flow with minimal loss to and from the roaming node. See policy management discussion below.

Figure 20 shows an example interface assisted roaming topology node data structure. Figure 20 shows this data structure implemented as a linked list, but it could alternatively be represented as an array where the next and previous fields are omitted. In a wireless network infrastructure, the "NPOA" may, for example, be the MAC address of the access point or base station that the mobile node is associated with. In other networks, it may be the unique identifier of an intervening network interconnect (e.g., gateway, IWF, etc.). The data structure may be seeded with static information or dynamically learned. Other information may also be associated with each node (e.g., MTU size, latency, cost, availability, etc.)

EXAMPLE FURTHER EMBODIMENT TO HANDLE CERTAIN RACE CONDITIONS

Through further experimentation evidence has shown that some network adapters may erroneously signal that they are (re)connected to the medium before they are totally registered on the network segment. In some instances during roaming events the storage area of where the network identifier is kept may not yet been updated, and thus it is possible for the

system to incorrectly believe that these adapters have roamed back onto the same subnet. Eventually, when the device finishes registering, the storage area is updated with the new network identifier, causing yet another ROAM signal to be generated. This scenario would correctly work if both pieces of information were gated together and only signaled once when the interface was finished registering with the network. However when polling it is difficult to determine when the network ID is valid if the "in contact with network" signal is generated previously.

In essences the roaming node may in fact be in media connectivity state since it can communicate at the media access level with the network, but in fact one cannot yet send any application data across the link since the registration process has not completed. Therefore, it is desirable to compensate for this condition. One way to provide such compensation is to determine peer connectivity by sending link confirmation frames, or what is more commonly known as an echo request/response packets. These echo or ping frames are generated by one peer (most likely the roaming node), to determine if two-way peer-to-peer connectivity is achievable. If the requesting peer receives a response frame to its request, it can be concluded that a duplex path has been achieved. At this point, the NPOA information can be regarded as valid until the next disconnect situation is realized. Other information, such as the reception of any frame from the peer on the interface in question, also allows the roaming node to assume the registration process has concluded and two-way communications is achievable.

Another race condition between the network interface and the underlying protocol stack situation has arisen that can sometimes cause a problem. It is possible for a device to have roamed to a new network segment and been signaled correctly from the interface below, but the

transport stack itself has not made the necessary adjustments to its routing table(s) for application data to flow. To compensate for this condition, an additional signal ROAM_SIGNAL_ROUTE_CHANGE, was added and is generated whenever the underlying transport's routing table changes. When this signal is indicated, the roaming subsystem clients take whatever action is necessary to determine if connectivity to the peer systems is achievable. This may entail the roaming client to enumerate through the underlying transport's routing table to determine if the routing modification has affected the communications path to the peer. Other more intrusive algorithms, such as the ones described above, can also be used to confirm that a two-way communication path exists between the peers.

Example Roaming Across Disjoint Networks

A further aspect of an example non-limiting preferred embodiment of our invention provides an algorithm and arrangement for accessing the MMS (Mobility Management Server) in what we call "disjoint networking" mode. The new algorithm allows for dynamic/static discovery of alternate network addresses that can be used to establish/continue communications with an MMS -- even in a disjoint network topology in which one network may have no knowledge of network addresses for another network.

In general, the algorithm allows for a list of alternate addresses that the MMS is available at to be forwarded to an MES (Mobile End System) during the course of a conversation. Thus, the MMS uses a connection over one network to send the MES one or more MMS network addresses or other MMS identities corresponding to other networks. As one example, this list can sent during circuit creation. It is also possible for the list to change midstream. In this case, the list can be updated at any time during the connection.

If/when the MES roams to another network, it uses the list of MMS "alias" addresses/identifications to contact the MMS from the new network point of attachment. This allows the MES to re-establish contact with the MMS over the new network connection even though the primary and ancillary networks may not share any address or other information.

Figure 21 shows a simplified flowchart of this new technique. Suppose that the MMS 102 is connected to two different disjoint networks or network segments N1 and N2. Suppose that the MES 104 is initially coupled to the MES 102 via network N1. Once a connection has been established between the MES 104 and the MMS 102 over network N1, the MMS 102 can send the MES 104 a list L of network addresses or other identifiers that the MMS is called on one or more other networks (e.g., network N2). The MES 104 receives and stores this list L. Then, when the MES 104 roams to another network (N2), it can access this stored list L and use it to efficiently re-establish communication with the MMS 102 over the new network (N2).

There are at least several uses for this new algorithm in addition to the ability to more efficiently obtain an alternative network address or other identifier for communicating with the MMS 102 over a disjoint network. One example usage is secure network operation. For example, using the algorithm shown in Figure 21, one can setup a secure network where the MMS 102 is used as a secure firewall/gateway from a multitude of networks (some/all may be wireless) and a corporate backbone, and allow for secure and seamless migration of the mobile node 104 between all disassociated networks. Think, for example, of the MMS 102 as a hub, with one fat pipe connecting to the corporate network and many little spokes connecting many logically discrete networks. Since they are logically discrete, traffic

on one network segment cannot reach another, except through the MMS 102 (which can act as a router in this example).

Normally for a node to roam from network segment to network segment, there must be routing information/paths provided on each network segment (i.e. default route, etc) specifying how to get back to the "main public or initial address" used to contact the MMS 102. Once a connection is established, that address is used for the life of the connection. When a frame is sent from the MES 104, the IP network (layer 3) infrastructure on the client and intermediary nodes (routers) looks at the destination address of the frame and correctly forwards the packet on to its ultimate destination (the MMS 102). This is done by using what is commonly referred to as IP forwarding, or IP routing. With this functionality turned on, frames (broadcasts, etc) from one network segment can leak onto another. By not using IP forwarding, frames sent on one segment are not forwarded onto the other, thus breaking the communications pipe or creating a disjoint network.

The alternate address list shown in Figure 21 has the effect of pushing or distributing some of the routing intelligence out to the MES 104. Each segment therefore can be kept discrete and without knowledge of any other segment attached to the MMS 102. The MES 104 can be authenticated by the MMS 102 so that the MMS only sends a list L to authorized MES units 104. When the MES 104 roams onto another networks segment, it can automatically select the correct address to use to initiate/continue communications with the MMS midstream, thus solving the disjoint network problem, and not require any changes to the routing infrastructure. This provides for a more secure computing environment by only letting validated users to gain access to the network.

For example, by using the MMS 102 in this manner combined with user level security/encryption, we can limit traffic from and to the corporate

backbone to only the frames destined for those nodes on that segment using the roaming techniques described above. Frames can be optionally encrypted to thwart any potential eavesdropping by devices that may be validated by the spoke network infrastructure.

Figure 22 shows an example. In Figure 22, the MMS 102 is attached to four separate and distinct networks (Ia, Ib, Ic, Id) without any interconnects or route information shared. For all intents and purposes, each network I is an island. Now envision an MES 104 being docked to one of the networks (e.g., Ic) using a wired connection on the corporate backbone. For example, suppose that the MES 104 acquires an address on the 192.168.x.x network to communicate with the MMS 102.

Now suppose that for some reason, the MES now needs to migrate or roam to the 10.1.x.x (Ia) network. Since the 10.1.x.x (Ia) network has no knowledge of the 192.168.x.x (Ib) network (i.e. no routes to it), when the MES 104 moves into its domain, the communication pipe is broken even though the MMS is attached to it. Again, the same thing happens when the mobile node 104 attaches to any of the other 10.x networks depicted.

Now using the algorithm shown in Figure 21, the MMS 102 at connection initiation time (or by some other method) shares its interfaces address on each of the various disjoint networks Ia, Ib, Ic, Id with the MES 104 and the MES records these. Once recorded, if the MES 104 roams into any one of the networks and detects that it has roamed onto a new network segment, the MES can now select the appropriate network address to communicate with the MMS for that network segment. If more then one address can be used, the MES 104 can select the appropriate address to use based on a number of metrics such as speed, cost, availability, hops, etc. An MES 104 that has not received a list as in Figure 21 may be effectively

prevented from roaming between the various networks because it has no way to contact the MMS over any network other than its "home" network.

Another application for the Figure 21 technique is in distributed network interfaces. In today's networks, folks have deployed what is known as Network Address Translators (NATs). By use of this conventional technology, one can have many network devices use only one public network address for access to information on the Internet. The technology provides this functionality by funneling all information and queries destined to the Internet through a single/few device(s). The device(s) records the request at the network layer, then remaps the address and port information in the packet to the devices own address/port tuple and sends it onto its destination. Upon reception of a frame from the Internet or other such network, the device(s) does the reverse look and forwards it back to the correct source by replacing its address/port tuple information with that of the initiating device. These mappings may be defined statically also at the NAT.

Suppose someone wants to use the MMS 102 for the LAN/WLAN internally and have it sit behind a NAT. Currently, unless the MMS 102 is the NAT, or by using a different proxy for all communications with the MMS, when someone roams outside of the bounds of the intranet, the MMS is no longer accessible since the address to converse with it is no longer accessible. With the Figure 21 algorithm, one can statically/dynamically define another interface address that is not directly attached to the MMS. Therefore, using the algorithm described above, the MES 104 can now automatically select the appropriate disjoint address to use when attaching to a network that is outside the intranet's domain.

Figure 23 illustrates this scenario. Suppose a node migrates from interface "d" to interface "g". Just supplying the MMS 102 local interfaces

would not allow access. The MES 104 needs a priori knowledge of the distributed interface. It can then select the necessary address to use on interface "g". The NAT 2000 will then do the appropriate translation of network address/port information on each packet to the internal interface "c" address. The reverse operation will happen on frames sent by the MMS 102 to the MES 104.

Example Policy Management and Location Based Services

A further non-limiting embodiment of the invention provides the unique ability to offer additional security, cost savings, and services based on a number of metrics. Since the MMS described above is intimately involved with each application session the MES establishes, either side (i.e., the MMS and/or the MES) can apply policy-based rules to tailor and control the communications between the MES and its ultimate peer. It can further condition or modify applications request based on the locale or proximity of the device and its attachment to the network. For example, the MMS and/or the MES can include a rules engine that applies learned, statically defined, or other rules based on policy to each application session that is established or request that is attempted. The MMS can further distribute some, none or part of such rules and/or processing to the MES to provide further metering or security against rogue attacks of the mobile device. Unlike certain other policy management technology available in a distributed topology, the MMS provides a central place to administer the rules and policy decisions and have them distributed to the remote device at any time during the course of a conversation/connection.

The rules themselves can be configured based on user, user group, device, device group, process, application identity and/or network point of attachment. Once defined (learned), they can be combined to govern and

control a variety of different events, activities, and/or services, including for example:

- denying, allowing or conditioning ingress access to the remote device;
- denying, allowing or conditioning access to specific network resources based on identity,
- denying, allowing or conditioning access to available or allowable bandwidth,
- denying, allowing or conditioning access to other network resources and/or
- modifying, conditioning or changing content or information.

Such decisions can be based on any of various different factors including for example:

- proximity, location, altitude and/or other characteristics of the mobile device,
- time of day,
- application or process identity, address, etc;
- application behavior (e.g., bandwidth requirements);
- current network conditions; and/or
- other static or dynamic factors

Furthermore by employing the distributed architecture, the MMS can also apply or share the same decision set. Having the MMS perform the policy management processing and/or decision making may be desirable in instances where the mobile device has limited processing power to execute the engine or bandwidth limitations are applicable, or for security purposes.

Figure 24 shows an example table of the some metrics (rules) that might be used to control a sample MES. This table may be populated either

statically or dynamically, and maybe updated anytime before, during, or after the connection/conversation. For example, a person could use a rules editor (e.g., a wizard) other mechanism to define entries in the table. In other example arrangements, the metrics could be automatically defined by the system based on learning, or could be dynamically changed based on changing conditions. The rules also have a priority assigned to them whether implied by the location in the table or specifically designated by an assignment. This priority allows the engine to correctly determine the expected behavior. Additional user interface functions allow the system administrator and or user of the device to interrogate the rules engine and test out the functionality of a given rule set.

The Figure 24 example table shows a number of example metrics on which policy management decisions may be based, including:

- MES communications capability (transmit only, receive only, or transmit and receive);
- Whether the MES request is proxied;
- MES source port;
- MES source address;
- MES destination port;
- MES destination address;
- MES protocol;
- Amount of bandwidth available;
- Process name(s), identities or other characteristics;
- Network name(s), identities or other characteristics;
- Location (e.g., GPS coordinates or other location information);
- Network point of attachment;
- User identity name, identity or other characteristic;

- Other metrics.

It will be appreciated that the invention should not be limited by the scope of the metrics entries in the example table as it is not meant to be an exhaustive list. The entries can be specific as in this example or use a generic mechanism (e.g., wildcards) to describe the desired behavior of the mobile node with regards to network access and entitlements.

The Figure 24 example table further includes a "deny request" entry that indicates the result of a policy management decision to be made based on the metrics. As one example, the particular example entries in the Figure 24 table specify that all connections to destination ports 20 and 21 should be denied or throttled back if the available bandwidth is reduced to less than 100,000 bytes per second. Furthermore, in the particular example shown, rules (rows) 3 and 4 allow only network traffic to flow to and from the MMS (all other network traffic that is not proxied is implicitly discarded).

In one example, before each RPC request or frame is processed, the rules engine is consulted to determine if the status of the operation. Base on the outcome of this process, the request may be allowed, denied or delayed. Figure 25 is an example flowchart of steps that may be performed by the MMS and/or the MIS to make policy management decisions.

Furthermore by combining the roaming technology outlined previously with other location or navigational information that mat be available, the MMS detects when a mobile end system has moved from one point of attachment to another. By combining this information in conjunction with the ability of the mobile end system to detect a change in environment of network topology, or locale enable the invention provides additional levels of location based monitoring and services.

To fully realize the potential of this information, enhancements to both the Internet Mobility Protocol and RPC engine are outlined. Several

new RPC protocol and configuration enhancements will be added to provide this functionality. These are listed below.

Example Location Change RPC

When the mobile end system has determined that it has moved to a new point of attachment using interface assisted roaming or some other method such as detecting changes from a global positioning system, it will send a formatted "Location Change RPC Request" message to its peer, in this case the mobility management server. The "Location Change RPC" formats one or more of the point of attachment identification information into a type, length, value format. The type identifies the kind of identification information, types supported will include but will not be limited to 48 bit IEEE MAC Addresses, IPV4 Addresses, IPV6 Addresses, longitude, latitude, altitude, and attachment names in ASCII. The length indicates the length in bytes of the identification data, and the data contains the actual point of attachment identification. The mobility management server upon receipt of the "Location Change RPC Request" will build a "Location Change Alert" that contains the point of attachment identification and other pertinent information such as the mobile end system identification, the user name, and PID. It then will forward the alert to the alert subsystem on the server. The alert will be formatted with the same type, length, data format utilized within the "Location Change RPC Request". The alert subsystem will then forward the location change alert with this information to all applications that have registered for the alert. Applications that have registered for the alert may include monitoring applications such as the current active status monitor, a long-term activity log, the policy management engine, and other third party applications and network management tools. One such third party application may combine

this location information with Web based maps to provide detailed information about a mobile end system's or MMS location. In addition to such applications, other actions can be associated with location change alerts. This includes sending an email, printing a message, launching a program and/or change in policy.

The Location Change RPC will contain a field in its header that indicates if it was triggered due to location change, distance change, or rate change.

In some instances, the MES may not know it has roamed. Depending on the medium and the network adapter it is attached to, the MMS may be the only entity that notices that the MES has migrated to a new point of attachment. Consider the case of a mobile router. The addresses behind the router stay the same, only the routers address changes. In this case, the MMS knows the new care of address of the MES. Therefore, for complete motion detection it needs to be a combination of the both the MES and MMS to detect motion. In the present embodiment, the MMS detects motion of the clients at the IMP layer when the source address changes and a new IMP message is received. When this occurs, the MMS locally generates a Location Change Alert. It also sends a message back to the MES that its point of attachment has changed.

Example Topology RPC

The "Topology RPC Request" is sent from the mobility management server to mobile end systems. Upon receipt of this RPC the mobile end system will read the topology information stored in its local data store and build a Topology RPC Response. The Topology RPC response will be formatted with a Total Length Field followed by consecutive type, length, data point of attachment identification followed by type, length, value data

indicating the subnet and network information. This information may be used on the server to build a complete topological map of the mobile network being served by the server.

Example Location information UI

The user interface on the server will provide a method for mapping and displaying location information. This location information will be available for each active mobile end system and the long-term activity log will maintain a history of all active and previously active mobile end system location changes. The user interface will permit the system administrator to configure the point of attachment information in human readable form. For example, if the point of attachment information is provided in the form of a 48-bit IEEE MAC address this MAC address will be displayed along with the information provided through the user interface on the server. If the point of attachment represented an access point in front of the "HallMark Cards" store it might be configured to present the following information "HallMark, Street Address, City, State, Zip". When displayed to the user, information "HallMark, Street Address, City, State, Zip" is presented.

Example Location RPC Timer

A configurable timer is provided on the mobile end system to limit the rate at which Location Change RPCs may be sent from the mobile end system to the mobility management server. If the timer interval is larger than the rate at which the point of attachment changes are occurring, the mobile end system will wait until the timer interval expires before generating another Location Change RPC.

Example Distance Change Notification

A distance metric will be provided for triggering the generation of Location Change RPCs. This setting configures the system to send an update when the user moves three dimensionally every n feet from, kilometer, or other appropriate unit of measure from the last point of origin. By default this setting is disabled. Enabling this setting causes a Change Notification when the distance interval in the configuration is exceeded.

Example Rate Threshold Notification

A rate change metric will be provided for triggering the generation of Location Change RPCs. This parameter is configured in distance per second such as miles per hour. It will specify an upper and lower bounds and a time interval that the attained rate must be sustained (i.e. 0 MPH for 10 minutes or 70 MPH for 1 minute). When this speed is reached a Location Change Notification will be generated.

EXAMPLES

The present invention finds application in a variety of real-world situations. For example:

Intermittently Connected Portable Computer

Many businesses have employees who occasionally telecommute or work from home. Such employees often use laptop computers to get their work done. While at work, the employees typically connect their laptop computers to a local area network such as an Ethernet through use of a docking port or other connector. The LAN connection provides access to network services (e.g., printers, network drives) and network applications (e.g., database access, email services).

Now suppose an employee working on a project needs to go home for the evening and wants to resume working from home. The employee can "suspend" the operating system and applications running on the laptop computer, pack up the laptop computer, and bring the laptop computer home.

Once home, the employee can "resume" the operating system and applications running on the laptop computer, and reconnect to the office LAN via a dialup connection and/or over the Internet. The Mobility Management Server (which continued to proxy the laptop computer vis-a-vis the network and its applications during the time the laptop computer was temporarily suspended) can re-authenticate the laptop computer and resume communicating with the laptop computer.

From the perspective of the employee now working from home, all of the network drive mappings, print services, email sessions, database queries, and other network services and applications, are exactly where the employee left them at the office. Furthermore, because the Mobility Management Service continued to proxy the laptop computer's sessions, none of those network applications terminated the laptop computer's sessions during the time the employee was traveling from the office to home. The invention thus provides efficient persistence of session across the same or multiple network mediums that is very powerful and useful in this and other contexts.

Mobile Inventory and Warehouse Application

Imagine a large warehouse or retail chain. Within this campus, inventory workers use vehicle mounted (i.e., trucks and forklifts) personal laptop computers and handheld data collection units and terminals to perform inventory management of goods. Warehouse and retail workers are

often inexperienced computer users that do not understand network sub-nets and require management supervision. The present invention allows the creation of a turnkey system that hides the complexity of the mobile network from the warehouse users. The users can move in and out of range of access points, suspend and resume their Mobile End Systems 104, and change locations without concern for host sessions, network addresses, or transport connections. In addition, the management software on the Mobility Management Server 102 provides management personnel with metrics such as number of transactions, which may be used to gauge worker productivity. Management can also use the network sub-net and access points to determine worker's last known physical location.

Mobile Medical Application

Imagine a large hospital using radio LAN technology for network communications between several buildings. Each building is on a unique sub-net. The present invention enables nurses and doctors to move from room to room with handheld personal computers or terminals -- reading and writing patient information in hospital databases. Access to the most recent articles on medication and medical procedures is readily available through the local database and the World Wide Web. While in the hospital, pagers (one and two way) are no longer required since the present invention allows continuous connection to the Mobile End System 104. Messages can be sent directly to medical personnel via the Mobile End System 104. As in the case with warehouse workers, medical personnel are not required to understand the mobile network they are using. In addition, the Mobile End System 104 allows medical personnel to disable radio transmission in area where radio emissions are deemed undesirable (e.g., where they might

interfere with other medical equipment) -- and easily resume and reconnect where they left off.

Trucking and Freight

Freight companies can use the present invention to track inventory. While docked at a warehouse, the Mobile End System 104 may use LAN technology to update warehouse inventories. While away from local services, the Mobile End System 104 can use Wide Area WAN services such as CDPD and ARDIS to maintain real time status and location of inventory. The Mobile End System 104 automatically switches between network infrastructures -- hiding the complexity of network topology from vehicle personnel.

Mobile Enterprise

Corporate employees may use the system in accordance with the present invention for access to E-mail, web content and messaging services while within an enterprise campus that has invested in an infrastructure such as 802.11. The cost of ownership is reduced since pager service and other mobile device services are no longer required. The purchase of mobile infrastructure is a one time capital expense as opposed to the costly "pay-per-use" model offered by many existing mobile device services.

IP Multiplication

If an organization has a LAN that needs to be connected to the Internet, the administrator of the LAN has two choices: get enough globally assigned addresses for all computers on the LAN, or get just a few globally assigned addresses and use the Mobility Management Server 102 in accordance with the present invention as an address multiplier. Getting a large number of IP addresses tends to be either expensive or impossible. A

small company using an Internet Service Provider (ISP) for access to the Internet can only use the IP addresses the ISP assigns – and the number of IP addresses limits the number of computers that can be on the Internet at the same time. An ISP also charges per connection, so the more computers that need to be on the Internet, the more expensive this solution becomes.

Using the Mobility Management Server 102 in accordance with the present invention as an address multiplier could solve many of these problems. The enterprise could put the Mobility Management Server 102 on hardware that is connected to the Internet via an ISP. Mobile End Systems 104 could then easily connect. Because all connection to the Internet would go through the Mobility Management Server 102, only one address from the ISP is required. Thus, using the present invention as an address multiplier allows the enterprise to get just a few (in many cases one) addresses and accounts from the ISP, and allows the entire LAN to have simultaneous connections to the Internet (assuming enough bandwidth is provided).

While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not to be limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the scope of the appended claims.

WHAT IS CLAIMED IS:

1. A mobile computing network including at least one mobile computing device coupled to the network via a network point of attachment, characterized by a policy-management arrangement that applies policy management rules based on various metrics including mobile computing device location.

2. A network as in claim 1 further characterized in that processing of the attributes of the rules can be distributed and applied at either the mobile computing device and or the mobility management server or both.

3. A network as in any of the preceding claims further characterized in that prioritization of the rules is either implied by position in the entry in such table or explicitly noted by an ordinal ensuring the expected behavior.

4. A network as in any of the preceding claims further characterized in that datastore for the rule attributes is locally or centrally administered via central management services

5. A network as in any of the preceding claims further characterized in that behavior of a particular application(s) is modified based on a number of metrics including cost of service, network point of attachment, trust relationship, etc.

6. A network as in any of the preceding claims further characterized in that the effect of the behavior modification is to allow, deny or delay a request based on attributes of the rules.

7. A network as in any of the preceding claims further characterized in that even if the application is already started, a rule or set of rules is invoked to modify the application(s) processes.

8. A network as in any of the preceding claims further characterized in that point of presence information (location) is further used to govern application behavior or provide relevant information to the mobile computing device.

9. A network as in any of the preceding claims further characterized in that rate of motion along with distance measurements is used to alter behavior of applications or the communication path.

10. A network as in any of the preceding claims further characterized in that topological information is extracted and displayed as result of the location information.

11. In a mobile computing network including at least one mobile computing device coupled to the network via a network point of attachment, an improvement comprising an interface-assisted roaming listener that detects, based at least in part on identification of the network point of attachment, whether said mobile computing device has roamed to a different network segment.

12. A network as in claim 11 wherein said mobile computing device includes a network interface adapter, and said listener obtains said network point of attachment identification from said network interface adapter.

13. A network as in claim 11 wherein said listener maintains a network topology map storing information that correlates said network point of attachment identification with further information concerning a network connection.

14. A network as in claim 11 wherein said listener detects when communications with said network is interrupted or reestablished.

15. A network as in claim 14 wherein said listener generates a roam signal in response to detection of (a) network communications interruption and re-establishment, and (b) change of said network point of attachment identification.

16. An interface-based listener for use in a mobile computing device, said interface-based listener integrating information from at least one network interface adapter with information available from at least one network stack to determine whether said mobile computing device has moved to a new network point of attachment.

17. The interface-based listener of claim 16 including a network topology map providing network connection information including network points of attachment information.

18. The listener of claim 17 wherein said listener dynamically constructs said network topology map based on learned information.

19. The interface-based listener of claim 16 including a status checker that checks status based on occurrence of an event.

20. The interface-based listener of claim 16 wherein said event comprises any of a timer timeout, a low level roaming driver callback, and a network level activity hint.

21. The interface-based listener of claim 16 including a connection information searcher that queries an interface as to whether the mobile computing system has already visited the current network point of attachment.

22. The interface-based listener of claim 16 including a connection arrangement that registers or reacquires a current address to be valid on a new network segment.

23. The interface-based listener of claim 16 including a roam signal generator that generates a roam signal in response to detection, based at least in part on information provided by an interface, that the mobile computing device has roamed to a different network segment.

24. The interface-based listener or claim 23 further including a heuristic analyzer that determines whether a previously assigned address is still valid.

25. A method of determining whether a mobile node has moved to a new network point of attachment, comprising:

(a) receiving network point of attachment identification information from a network interface;

(b) using said network point of attachment identification information to determine whether said mobile node has moved to a new network point of attachment; and

(c) generating signaling in response to said step (b).

26. A method as in claim 25 further including maintaining a network topology map, and using said map to perform step (c).

27. A method as in claim 25 wherein said step (c) includes generating a roam signal.

28. A method as in claim 25 wherein said step (b) includes obtaining said network point of attachment information from a network adapter.

29. A method as in claim 25 further including falling back to an alternative roaming detection mechanism if a network interface is not available that supports generic signaling.

30. A method as in claim 25 further including selecting, at least in part in response to said network point of attachment information, between alternate network connection paths.

31. A method for facilitating communication with a mobile system over disjoint networks comprising:

establishing communications between a node and said mobile system over a first network;

sending to the mobile system over the first network, data identifying the node on at least a second network disjoint from the first network; and

using the data to establish communication between the mobile system and the node over the second network.

32. The method of claim 31 further including authenticating the mobile system for authorization to communicate with the node over the second network before sending the data to the node over the first network.

33 The method of claim 21 wherein the sending step comprises sending distributed interface data to the mobile system over the first network.

34. A network as in claim 12 wherein said mobile computing device network interface adapter is physically attached to said network.

35. A network as in claim 12 wherein said mobile computing device communicates wirelessly with the network point of attachment.

36. A method for maintaining communication between a mobile computing system and a network node as the mobile computing system roams between over plural disjoint networks comprising:

establishing communications between the mobile system and a node via a first network segment;

sending the mobile computing system, via the first network segment, information for use in re-establishing communications with said node via plural further network segments each of which are disjoint from the first network segment; and

using said information to re-establish communications between the mobile computing system and the node via any of said plural further, disjoint network segments.

37. The process of claim 36 wherein said information comprises distributed interface data.

38. A process for providing least cost routing in a network having plural disjoint segments, comprising:

(a) establishing communications between the network and a temporarily-attached mobile computing device;

(b) using a roaming mechanism to allow the temporarily-attached mobile computing device to roam between said plural disjoint segments; and

(c) enforcing at least one policy parameter to enable efficient automatic selection of alternate valid network paths for re-establishing communication between the network and the mobile computing device in response to mobile computing device roaming.

39. The process of claim 38 wherein the policy parameter comprises an element selected from the following group: bandwidth, cost per data unit and quality of service.

40. In a mobile computing network including at least one peer computing system and at least one mobile computing device coupled to the network via a physical link, an improvement comprising a server coupled to the network, said server proxying communications between the mobile computing device and the peer computing system so as to maintain a continuous virtual data stream connection between the mobile computing device and the peer computing system during times when the physical link to the mobile computing device is temporarily interrupted.

41. A network as in claim 40 wherein said mobile computing device has a point-of-presence address on said network, said peer computing

system communicates with said server using a virtual address, and said server maps said virtual address to said point-of-presence address.

42. A network as in claim 41 wherein said server detects when said mobile computing device has changed its point-of-presence address, and re-maps said virtual address to said changed point-of-presence address.

43. A network as in claim 40 wherein said server queues and responds to requests from said peer computing system on behalf of said mobile computing device during times when said mobile computing device is temporarily unreachable or roaming.

44. A network as in claim 40 wherein said server communicates with said mobile computing device using a conventional transport protocol.

45. A network as in claim 44 wherein said server communicates with said mobile computing device using remote procedure calls.

46. A network as in claim 44 wherein said server communicates with said mobile computing device using an Internet Mobility Protocol.

47. A network as in claim 46 wherein said Internet Mobility Protocol provides for automatic removal of datagrams based on user-configurable timeouts.

48. A network as in claim 46 wherein said Internet Mobility Protocol provides for automatic removal of datagrams based on user-configurable retries.

49. A network as in claim 40 wherein said server performs per-user policy management of consumption of network resources by said mobile computing device.

50. A network as in claim 40 wherein said server provides user-configurable session priorities for said sessions of said mobile computing device.

51. A network as in claim 40 wherein said mobile network includes plural sub-networks, and said mobile computing device uses Dynamic Host Configuration Protocol along with other methodologies to allow said mobile computing device to roam from one of said plural sub-networks to another of said plural sub-networks.

52. A network as in claim 40 wherein said server comprises a Mobility Management Server.

53. A network as in claim 40 further including at least one mobile interconnect coupling said mobile computing devices to said server.

54. A method of maintaining a persistent connection with at least one mobile computing device in a mobile computing environment, said method including:

managing at least one session between said mobile computing device and at least one further computing device, and

maintaining the session when the mobile computing device becomes unreachable, suspends or changes network address.

55. A method as in claim 54 further including providing at least one user configurable session priority for said session.

56. A method as in claim 54 wherein said managing step includes managing consumption of network resources by said mobile computing device.

57. A method as in claim 54 wherein the mobile computing environment includes plural sub-networks, and said maintaining step uses Dynamic Host Configuration Protocol to maintain the session when said mobile computing device roams between said sub-networks.

58. A method as in claim 54 wherein said managing step communicates datagrams with said mobile computing device and automatically removes unreliable ones of said datagrams based on at least one user configurable parameter.

59. A method as in claim 58 wherein said user configurable parameter comprises a timeout.

60. A method as in claim 58 wherein said user configurable parameter comprises a user configurable retry number.

61. A method as in claim 54 further including providing said mobile computing device with a variable point of presence address, and wherein said managing step includes mapping said variable point of presence address to a virtual address, the session being associated with the virtual address.

62. A method as in claim 54 wherein said managing step includes using a Remote Procedure Call protocol to communicate with the mobile computing device.

63. A method as in claim 54 wherein said maintaining step maintains the connection state of said session during interruptions in a physical link connecting said mobile computing device with said mobile computing environment.

64. A method as in claim 54 wherein said managing step includes communicating with said mobile computing device using at least one standard transport protocol.

65. A method as in claim 54 wherein said mobile computing device includes plural application sources, and said managing step includes coalescing data from said plural application sources into a stream, and forwarding said stream.

66. A method as in claim 65 further including demultiplexing said coalesced data from said stream and forwarding said demultiplexed data to plural associated destinations.

67. A method as in claim 65 wherein said stream includes frames, and said coalescing includes dynamically resizing said frames to accommodate a maximum transmission unit of the mobile computing environment.

68. A method as in claim 65 wherein said coalescing includes maintaining semantics of unreliable data, and selecting discarding said unreliable data based on said semantics.

69. A method as in claim 54 wherein said managing step includes providing guaranteed delivery of messages to and/or from said mobile computing device.

70. A method as in claim 54 wherein said managing step includes controlling which network resources are accessible by said mobile computing device.

71. A server for maintaining a persistent connection with at least one mobile computing device in a mobile computing environment including at least one further computing device, said server including:

a session manager that manages at least one session between said mobile computing device and said at least one further computing device, said session manager maintaining the session when the mobile computing device becomes unreachable, suspends or changes network address.

72. A server as in claim 71 wherein said session manager includes a session priority queue that provides at least one user configurable session priority for said session.

73. A server as in claim 71 wherein said session manager includes means for managing consumption of network resources by said mobile computing device.

74. A server as in claim 71 wherein the mobile computing environment includes plural sub-networks, and said session manager uses Dynamic Host Configuration Protocol to maintain the session when said mobile computing device roams between said sub-networks.

75. A server as in claim 71 wherein said session manager communicates datagrams with said mobile computing device and automatically removes unreliable ones of said datagrams based on at least one user configurable parameter.

76. A server as in claim 75 wherein said user configurable parameter comprises a timeout.

77. A server as in claim 75 wherein said user configurable parameter comprises a user configurable retry number.

78. A server as in claim 71 wherein said mobile computing environment provides said mobile computing device with a variable point of presence address, and said session manager maps said variable point of presence address to a virtual address, the session being associated with the virtual address.

79. A server as in claim 71 wherein said session manager uses a Remote Procedure Call protocol to communicate with the mobile computing device.

80. A server as in claim 71 wherein said mobile computing environment includes at least one physical link connecting said mobile computing device with said mobile computing environment, and said session manager maintains the connection state of said session during interruptions in said physical link.

81. A server as in claim 71 wherein session manager communicates with said mobile computing device using at least one standard transport protocol.

82. A server as in claim 71 wherein said mobile computing device includes plural application sources, and said session manager coalesces data associated with said plural application sources into a stream, and forwards said stream.

83. A server as in claim 71 wherein said mobile computing device includes plural application sources, and said session manager demultiplexes coalesced data from said plural application sources and forwards said demultiplexed data to plural associated destinations.

84. A server as in claim 71 wherein session manager communicates with said mobile computing device using frames, and dynamically resizes said frames to accommodate a maximum transmission unit of the mobile computing environment.

85. A server as in claim 71 wherein said session manager maintains semantics of unreliable data, and selectively discards said unreliable data based on said semantics.

86. A server as in claim 71 wherein said session manager provides guaranteed delivery of messages to and/or from said mobile computing device.

87. A server as in claim 71 wherein said session manager places controls on mobile computing environment resources said mobile computing device can access.

88. In a mobile computing environment including a proxy server, a mobile computing device that maintains a persistent virtual connection with at least one further computing device during times when the mobile computing device becomes unreachable, suspends or changes network address, said mobile computing device including:

a transport driver interface, and

a mobile interceptor coupled to said transport driver interface, said mobile interceptor intercepting requests for network services at said transport driver interface, generating Remote Procedure Calls responsive to said requests for network services, and forwarding said Remote Procedure Calls to said proxy server.

89. A mobile computing device as in claim 88 wherein said mobile interceptor includes a session priority queue that provides at least one user configurable session priority.

90. A mobile computing device as in claim 88 wherein said mobile interceptor includes means for managing consumption of network resources by said mobile computing device.

91. A mobile computing device as in claim 88 wherein the mobile computing environment includes plural sub-networks, and the mobile computing device further includes means for using Dynamic Host Configuration Protocol to obtain a point of presence address when said mobile computing device roams between said sub-networks.

92. A mobile computing device as in claim 88 wherein said mobile interceptor communicates datagrams with proxy server and automatically removes unreliable ones of said datagrams based on at least one user configurable parameter.

93. A mobile computing device as in claim 92 wherein said user configurable parameter comprises a timeout.

94. A mobile computing device as in claim 92 wherein said user configurable parameter comprises a user configurable retry number.

95. A mobile computing device as in claim 88 wherein said mobile computing device has an associated a variable point of presence address that said Mobility Management Server maps to a virtual address.

96. A mobile computing device as in claim 88 wherein said mobile interceptor uses a Remote Procedure Call protocol to communicate with the said Mobility Management Server.

97. A mobile computing device as in claim 88 wherein said mobile computing environment includes at least one physical link connecting said mobile computing device with said mobile computing environment, and said mobile interceptor receives updated connection state information of at least one session from said Mobility Management Server after an interruption in said physical link.

98. A mobile computing device as in claim 88 wherein said mobile computing device includes a standard transport protocol handler, and said mobile interceptor communicates with said Mobility Management Server via said standard transport protocol handler.

99. A mobile computing device as in claim 88 wherein said mobile computing device includes plural application sources, and said mobile interceptor coalesces data associated with said plural application sources into a stream, and forwards said stream to said Mobility Management Server.

100. A mobile computing device as in claim 88 wherein said mobile computing device includes plural application destinations, mobile interceptor demultiplexes coalesced data from plural application sources and forwards said demultiplexed data to said plural application destinations.

101. A mobile computing device as in claim 88 wherein mobile interceptor communicates with said proxy server using frames, and

dynamically resizes said frames to accommodate a maximum transmission unit of the mobile computing environment.

102. A mobile computing device as in claim 88 wherein said mobile interceptor maintains semantics of unreliable data, and selectively discards said unreliable data based on said semantics.

103. A mobile computing device as in claim 88 wherein said mobile interceptor provides guaranteed delivery of messages to and/or from said proxy server.

104. A mobile computing device as in claim 88 wherein said mobile interceptor places controls on mobile computing environment resources said mobile computing device can access.

105. A mobile computing environment comprising:

at least one mobile computing device including:

a transport driver interface, and

a mobile interceptor coupled to said transport driver interface, said mobile interceptor intercepting requests for network services at said transport driver interface, generating Remote Procedure Calls responsive to said requests for network services, and forwarding said Remote Procedure Calls to at least one proxy server;

said proxy server including at least one work dispatcher that receives and handles said Remote Procedure Calls forwarded by said mobile interceptor, said proxy server including a proxy queue that proxies a virtual session on behalf of said mobile computing device when the mobile computing device becomes temporarily disconnected from said mobile computing environment.

106. In a mobile computing network including at least one mobile computing device coupled to the network via a network point of attachment, an improvement comprising an interface-assisted roaming listener that detects, based at least in part on identification of the network point of attachment, whether said mobile computing device has roamed to a different network segment.

107. A network as in claim 106 wherein said mobile computing device includes a network interface adapter, and said listener obtains said network point of attachment identification from said network interface adapter.

108. A network as in claim 106 wherein said listener maintains a network topology map storing information that correlates said network point of attachment identification with further information concerning a network connection.

109. A network as in claim 106 wherein said listener detects when communications with said network is interrupted or reestablished.

110. A network as in claim 109 wherein said listener generates a roam signal in response to detection of (a) network communications interruption and re-establishment, and (b) change of said network point of attachment identification.

111. An interface-based listener for use in a mobile computing device, said interface-based listener integrating information from at least one network interface adapter with information available from at least one network stack to determine whether said mobile computing device has moved to a new network point of attachment.

112. The interface-based listener of claim 111 including a network topology map providing network connection information including network points of attachment information.

113. The listener of claim 112 wherein said listener dynamically constructs said network topology map based on learned information.

114. The interface-based listener of claim 111 including a status checker that checks status based on occurrence of an event.

115. The interface-based listener of claim 111 wherein said event comprises any of a timer timeout, a low level roaming driver callback, and a network level activity hint.

116. The interface-based listener of claim 111 including a connection information searcher that queries an interface as to whether the mobile computing system has already visited the current network point of attachment.

117. The interface-based listener of claim 111 including a connection arrangement that registers or reacquires a current address to be valid on a new network segment.

118. The interface-based listener of claim 111 including a roam signal generator that generates a roam signal in response to detection, based at least in part on information provided by an interface, that the mobile computing device has roamed to a different network segment.

119. The interface-based listener or claim 118 further including a heuristic analyzer that determines whether a previously assigned address is still valid.

120. A method of determining whether a mobile node has moved to a new network point of attachment, comprising:

(a) receiving network point of attachment identification information from a network interface;

(b) using said network point of attachment identification information to determine whether said mobile node has moved to a new network point of attachment; and

(c) generating signaling in response to said step (b).

121. A method as in claim 120 further including maintaining a network topology map, and using said map to perform step (c).

122. A method as in claim 120 wherein said step (c) includes generating a roam signal.

123. A method as in claim 120 wherein said step (b) includes obtaining said network point of attachment information from a network adapter.

124. A method as in claim 120 further including falling back to an alternative roaming detection mechanism if a network interface is not available that supports generic signaling.

125. A method as in claim 120 further including selecting, at least in part in response to said network point of attachment information, between alternate network connection paths.

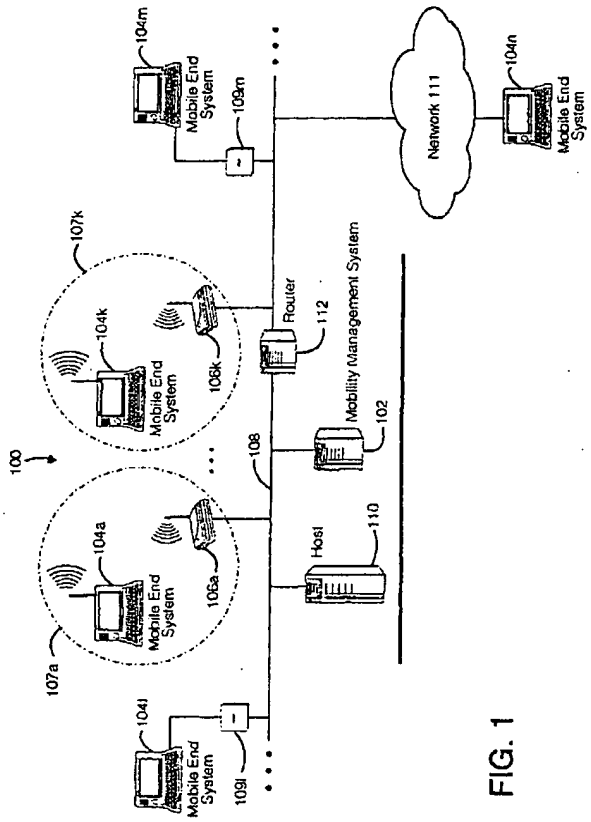

100
104m
Mobile End System
109m
Network 111
104n
Mobile End System
107k
104k
Mobile End System
106k
Router
112
Mobility Management System
102
108
104a
Mobile End System
106a
Host
110
107a
104l
Mobile End System
109l

FIG. 1

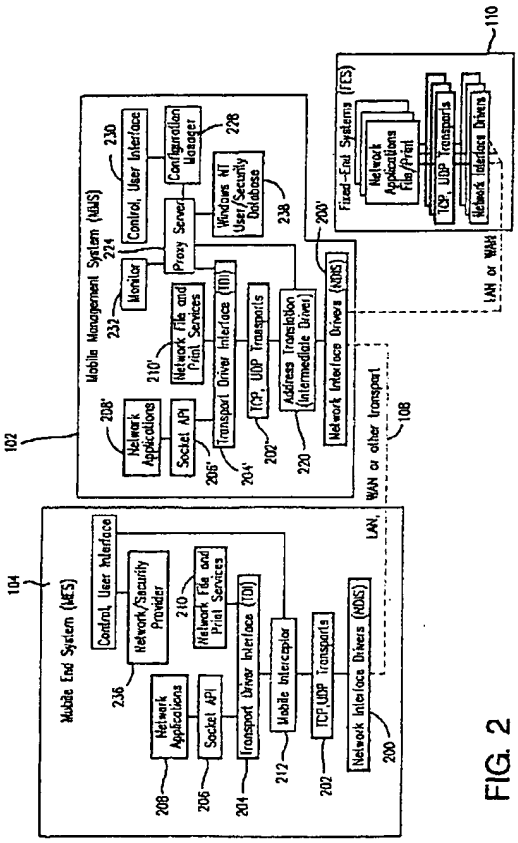

Mobile End System (MES)
Control, User Interface
Network/Security Provider
Network File and Print Services
Network Applications
Socket API
Transport Driver Interface (TDI)
Mobile Interceptor
TCP, UDP Transports
Network Interface Drivers (NDIS)
LAN, WAN or other transport
Mobile Management System (MMS)
Control, User Interface
Configuration Manager
Proxy Server
Windows NT User/Security Database
Monitor
Network File and Print Services
Transport Driver Interface (TDI)
TCP, UDP Transports
Address Translation (Intermediate Driver)
Network Interface Drivers (NDIS)
Network Applications
Socket API
Fixed-End Systems (FES)
Network Applications File/Print
TCP, UDP Transports
Network Interface Drivers
LAN or WAN
102
104
108
110
200
200'
202
204
204'
205
205'
206
206'
208
208'
210
210'
212
220
230
232
234
236
238

FIG. 2

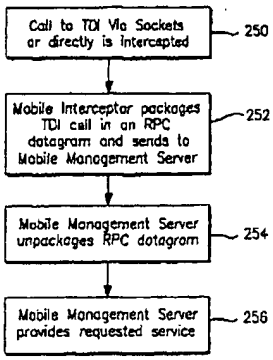

Call to TDI Via Sockets or directly is intercepted
250
Mobile Interceptor packages TDI call in an RPC datagram and sends to Mobile Management Server
252
Mobile Management Server unpackages RPC datagram
254
Mobile Management Server provides requested service
256

FIG. 2A

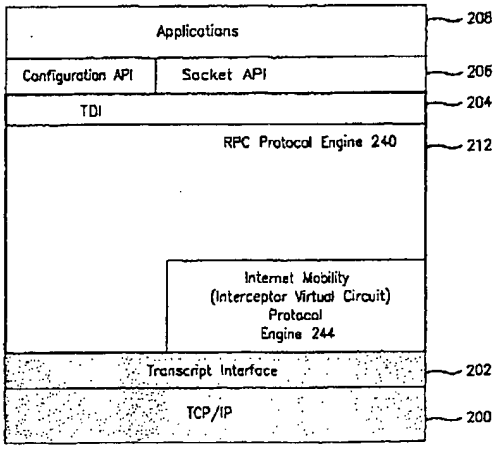

Applications
208
Configuration API
Socket API
206
TDI
204
RPC Protocol Engine 240
212
Internet Mobility (Interceptor Virtual Circuit) Protocol Engine 244
Transcript Interface
202
TCP/IP
200

FIG. 3

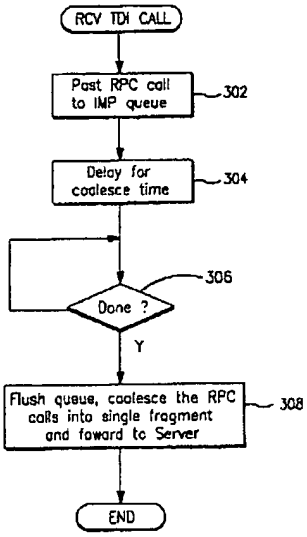

RCV TDI CALL
Past RPC call to IMP queue
302
Delay for coalesce time
304
306
Done ?
Y
Flush queue, coalesce the RPC calls into single fragment and foward to Server
308
END

FIG. 3A

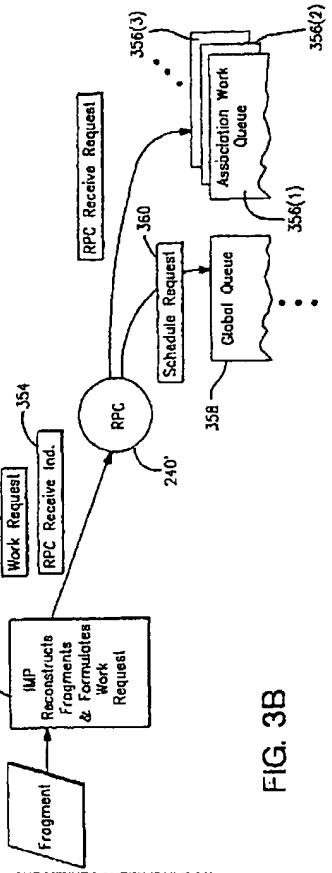

356(3)
356(2)
Association Work Queue
356(1)
RPC Receive Request
360
Schedule Request
Global Queue
358
354
RPC
240'
352
Work Request
RPC Receive Ind.
244'
IMP Reconstructs Fragments & Formulates Work Request
Fragment

FIG. 3B

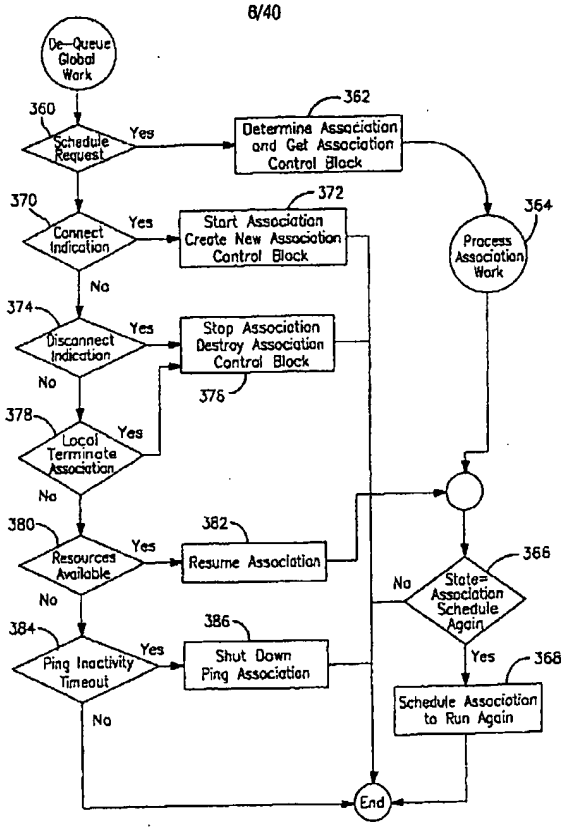

De-Queue Global Work
360
Schedule Request
Yes
362
Determine Association and Get Association Control Block
364
Process Association Work
370
Connect Indication
Yes
372
Start Association Create New Association Control Block
No
374
Disconnect Indication
Yes
Stop Association Destroy Association Control Block
376
No
378
Local Terminate Association
Yes
No
380
Resources Available
Yes
382
Resume Association
No
366
State= Association Schedule Again
No
384
Ping Inactivity Timeout
Yes
386
Shut Down Ping Association
Yes
368
Schedule Association to Run Again
No
End

FIG. 4

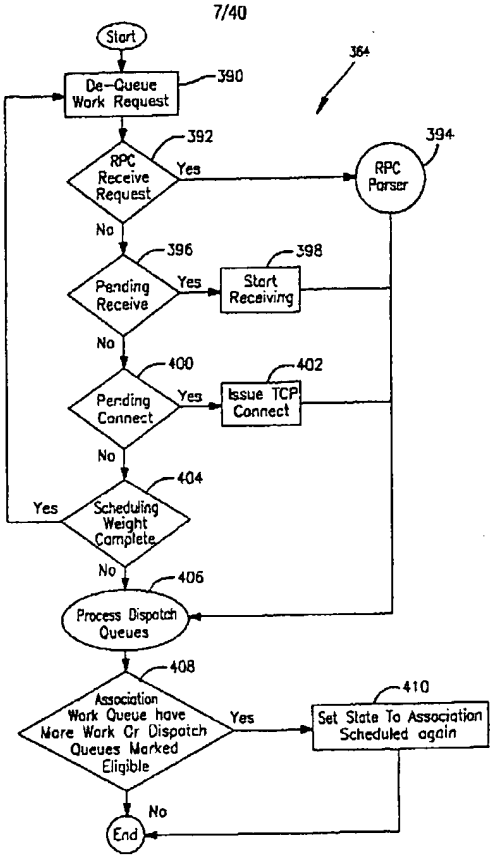


FIG. 5 Process Association Work

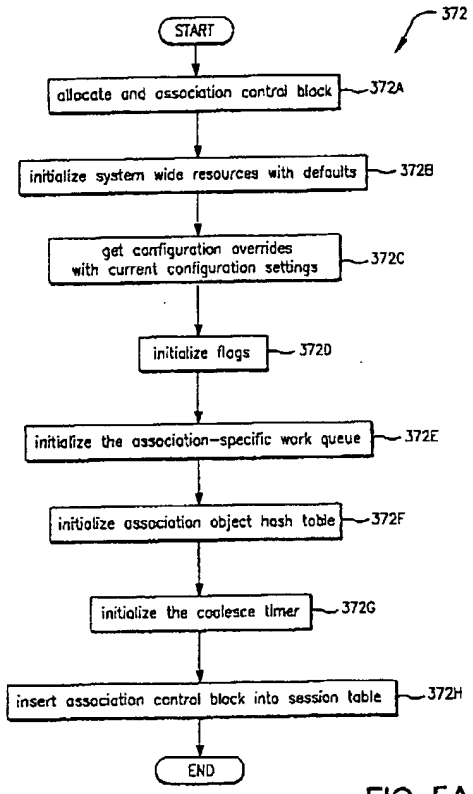

372
START
allocate and association control block
372A
initialize system wide resources with defaults
372B
get configuration overrides with current configuration settings
372C
initialize flags
372D
initialize the association-specific work queue
372E
initialize association object hash table
372F
initialize the coalesce timer
372G
insert association control block into session table
372H
END

FIG. 5A

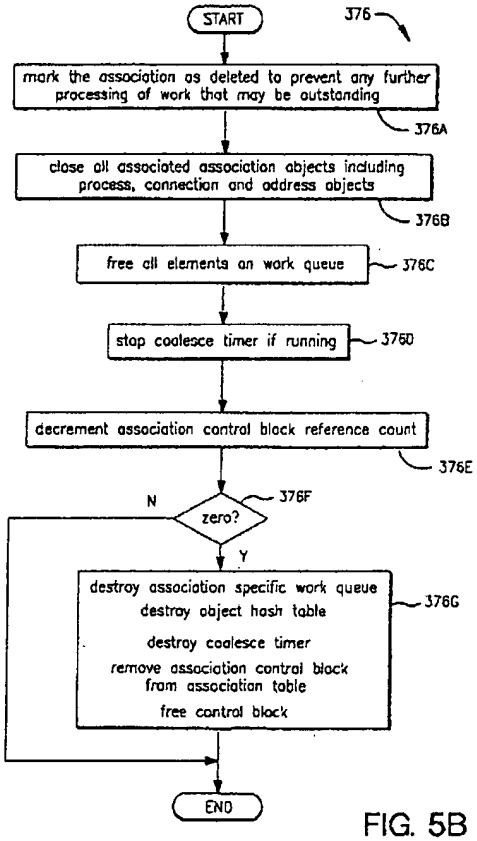

START
376
mark the association as deleted to prevent any further processing of work that may be outstanding
376A
close all associated association objects including process, connection and address objects
376B
free all elements on work queue
376C
stop coalesce timer if running
376D
decrement association control block reference count
376E
N
zero?
376F
Y
destroy association specific work queue
destroy object hash table
destroy coalesce timer
remove association control block from association table
free control block
376G
END

FIG. 5B

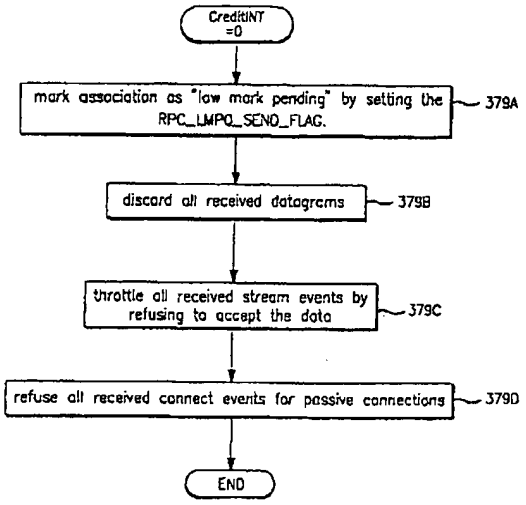

CreditINT =0
mark association as "low mark pending" by setting the RPC_LMPO_SEND_FLAG.
379A
discard all received datagrams
379B
throttle all received stream events by refusing to accept the data
379C
refuse all received connect events for passive connections
379D
END

FIG. 5C

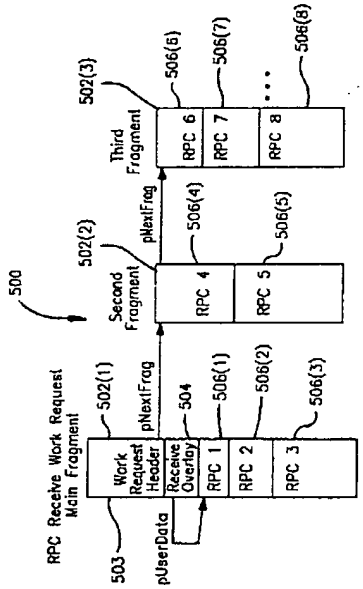

500
RPC Receive Work Request Main Fragment
Work Request Header
Receive Overlay
RPC 1
RPC 2
RPC 3
502(1)
pNextFrag
504
506(1)
506(2)
506(3)
503
pUserData
502(2)
Second Fragment
RPC 4
RPC 5
506(4)
506(5)
pNextFrag
502(3)
Third Fragment
RPC 6
RPC 7
RPC 8
506(6)
506(7)
506(8)

FIG. 6

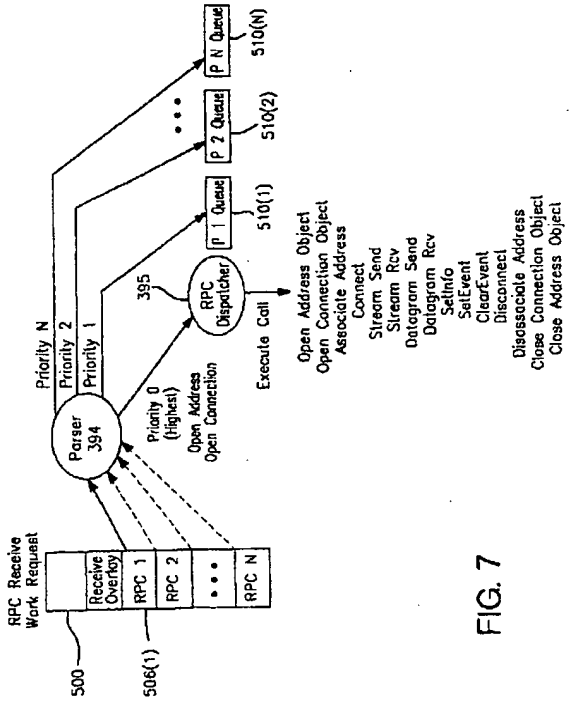

RPC Receive Work Request
Receive Overlay
RPC 1
RPC 2
RPC N
500
506(1)
Parser 394
Priority N
Priority 2
Priority 1
Priority 0 (Highest)
Open Address
Open Connection
395
RPC Dispatcher
P 1 Queue
P 2 Queue
P N Queue
510(1)
510(2)
510(N)
Execute Call
Open Address Object
Open Connection Object
Associate Address
Connect
Stream Send
Stream Rcv
Datagram Send
Datagram Rcv
SetInfo
SetEvent
ClearEvent
Disconnect
Disassociate Address
Close Connection Object
Close Address Object

FIG. 7

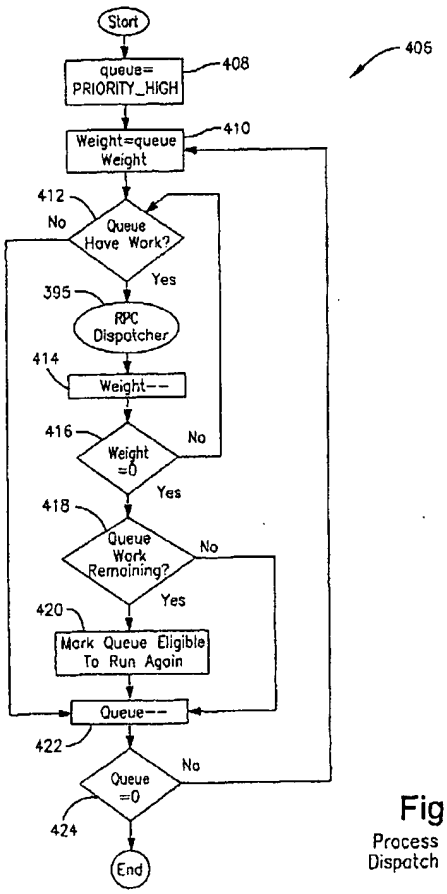


Fig. 8
Process Priority Dispatch Queues

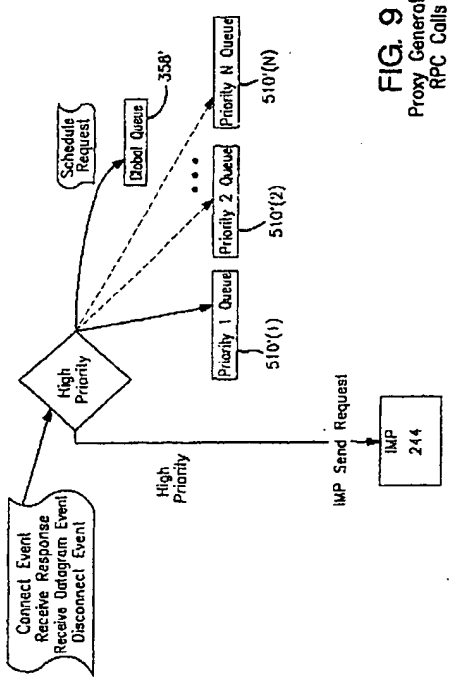
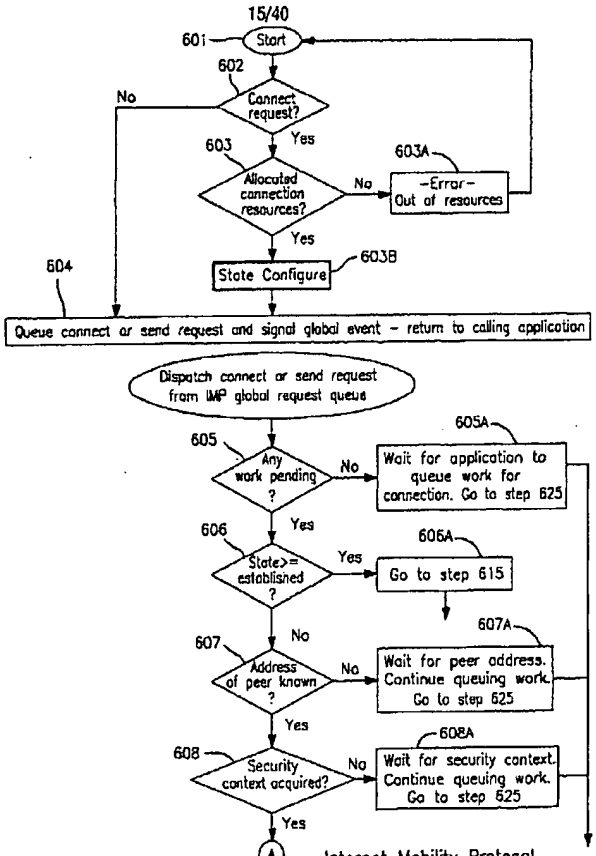


FIG. 9
Proxy Generated RPC Calls



Internet Mobility Protocol Connection Decision Tree.

FIG. 10A Connect and Send request logic

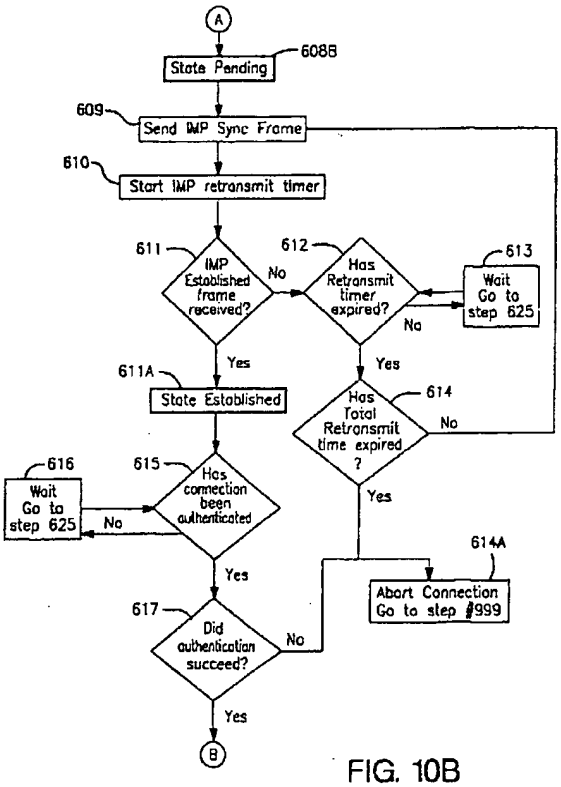

A
State Pending
608B
609
Send IMP Sync Frame
610
Start IMP retransmit timer
611
IMP Established frame received?
No
612
Has Retransmit timer expired?
613
Wait Go to step 625
No
Yes
611A
State Established
Yes
614
Has Total Retransmit time expired ?
No
616
Wait Go to step 625
615
Has connection been authenticated
No
Yes
614A
Abort Connection Go to step #999
617
Did authentication succeed?
No
Yes
B

FIG. 10B

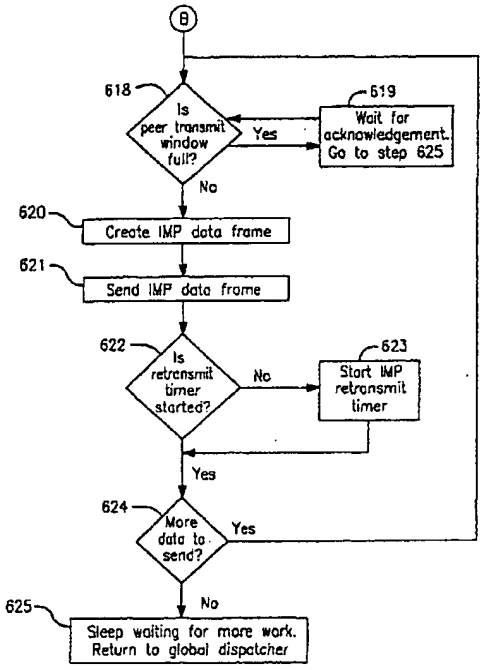

B
618
Is peer transmit window full?
Yes
619
Wait for acknowledgement. Go to step 625
No
620
Create IMP data frame
621
Send IMP data frame
622
Is retransmit timer started?
No
623
Start IMP retransmit timer
Yes
624
More data to send?
Yes
No
625
Sleep waiting for more work. Return to global dispatcher

FIG. 10C

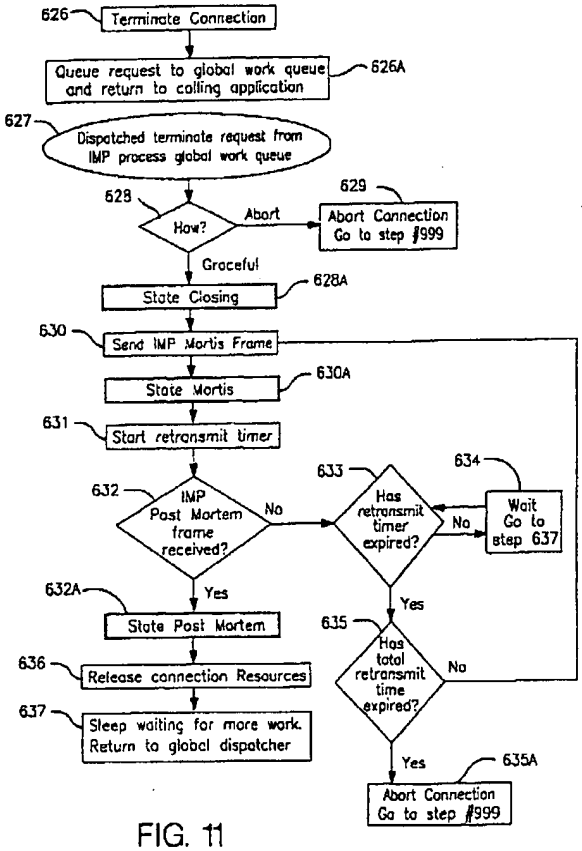


FIG. 11

Terminate Connection request logic

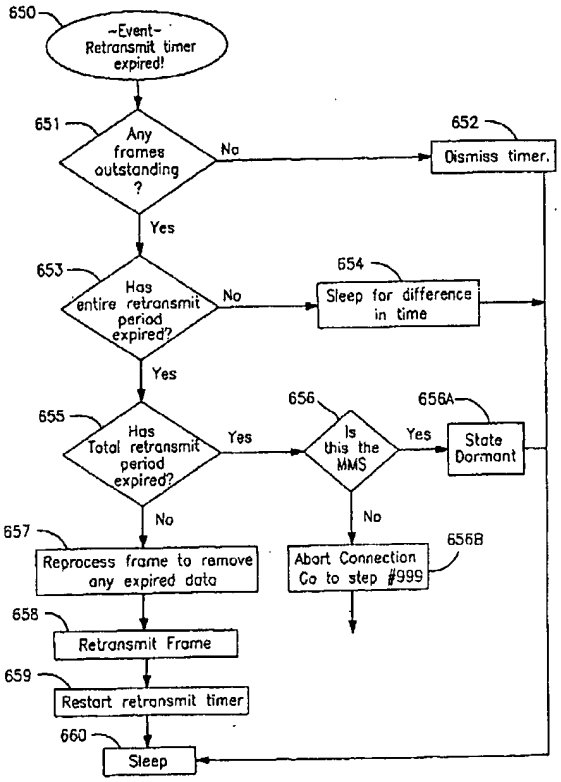


Fig. 12

Retransmit Event Logic

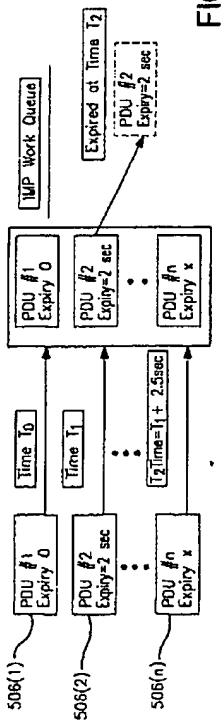


FIG. 12A

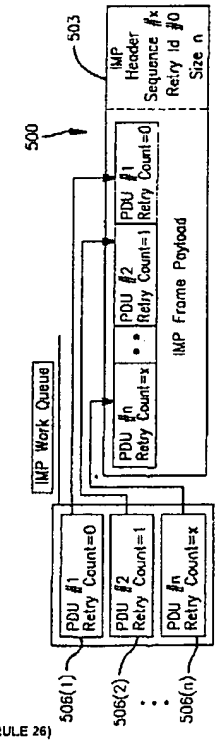


FIG. 12B

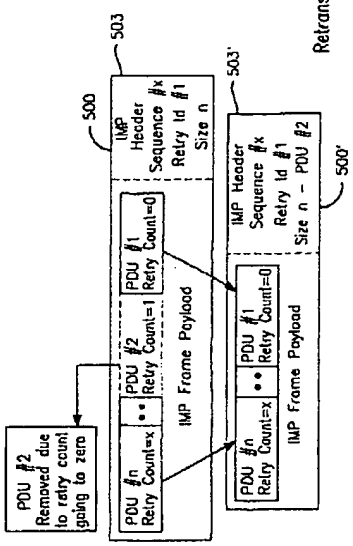


FIG. 12C
Retransmission of IMP Frame

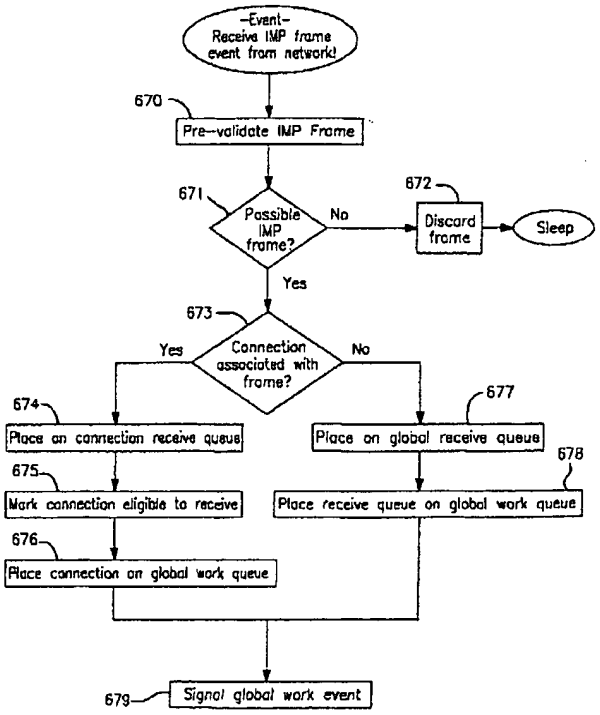


FIG. 13A

Receive Event Logic

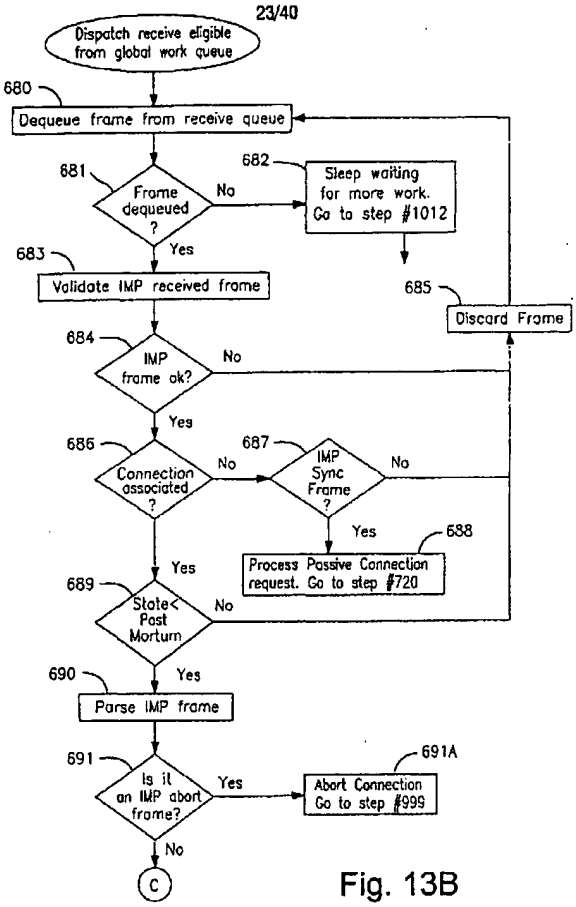

Dispatch receive eligible from global work queue
680
Dequeue frame from receive queue
681
Frame dequeued ?
No
682
Sleep waiting for more work. Go to step #1012
Yes
683
Validate IMP received frame
685
Discard Frame
684
IMP frame ok?
No
Yes
686
Connection associated ?
No
687
IMP Sync Frame ?
No
Yes
688
Process Passive Connection request. Go to step #720
Yes
689
State< Post Mortum
No
Yes
690
Parse IMP frame
691
Is it an IMP abort frame?
Yes
691A
Abort Connection Go to step #999
No
C

Fig. 13B

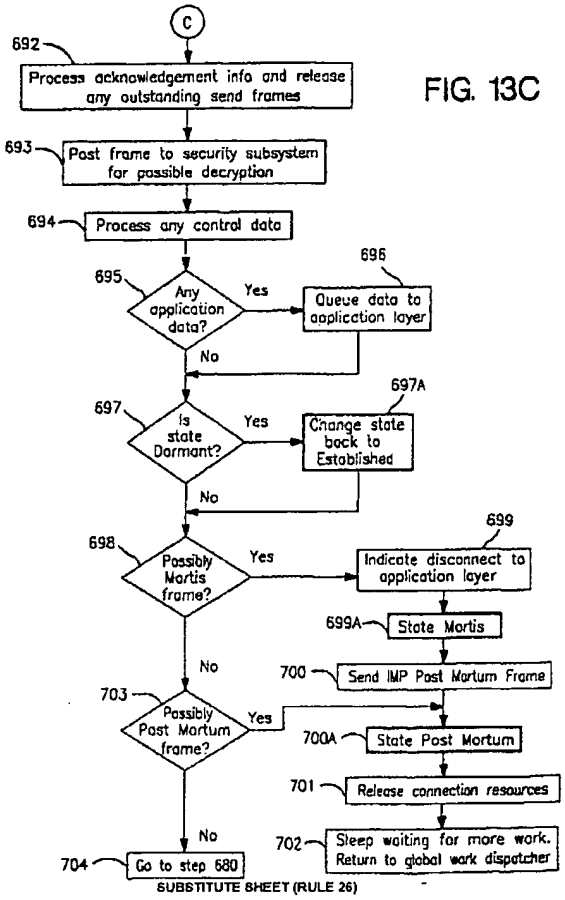

C
692
Process acknowledgement info and release any outstanding send frames
693
Post frame to security subsystem for possible decryption
694
Process any control data
695
Any application data?
Yes
696
Queue data to application layer
No
697
Is state Dormant?
Yes
697A
Change state back to Established
No
698
Possibly Mortis frame?
Yes
699
Indicate disconnect to application layer
699A
State Mortis
No
700
Send IMP Post Mortum Frame
703
Possibly Post Mortum frame?
Yes
700A
State Post Mortum
701
Release connection resources
No
702
Sleep waiting for more work. Return to global work dispatcher
704
Go to step 680

FIG. 13C

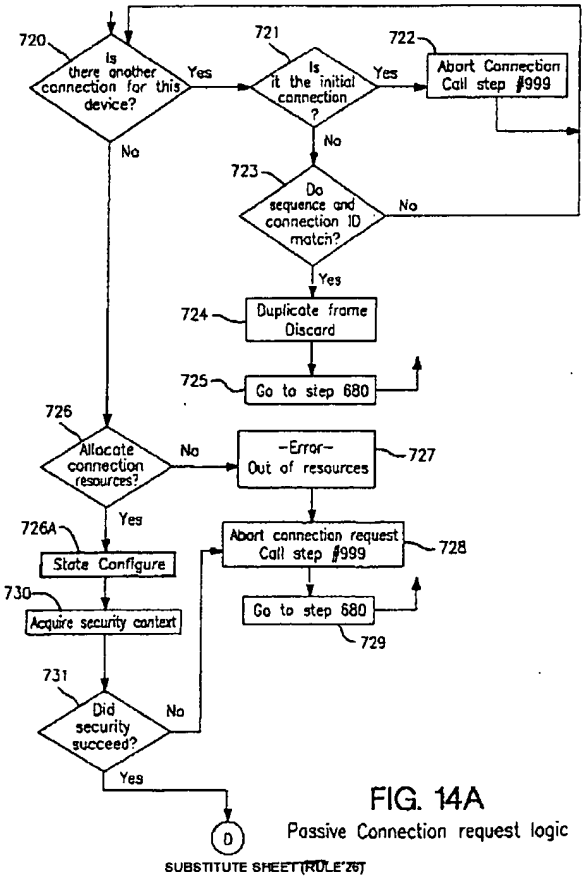


FIG. 14A
Passive Connection request logic

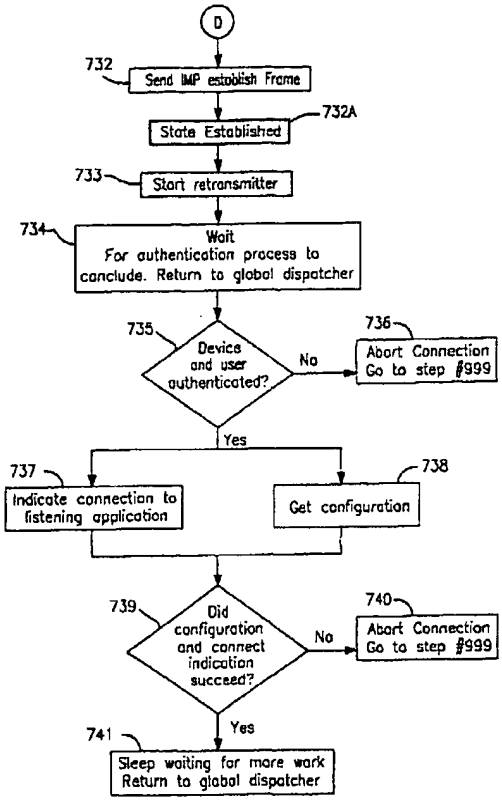

D
732
Send IMP establish Frame
732A
State Established
733
Start retransmitter
734
Wait
For authentication process to conclude. Return to global dispatcher
735
Device and user authenticated?
No
736
Abort Connection
Go to step #999
Yes
737
Indicate connection to listening application
738
Get configuration
739
Did configuration and connect indication succeed?
No
740
Abort Connection
Go to step #999
Yes
741
Sleep waiting for more work
Return to global dispatcher

FIG. 14B

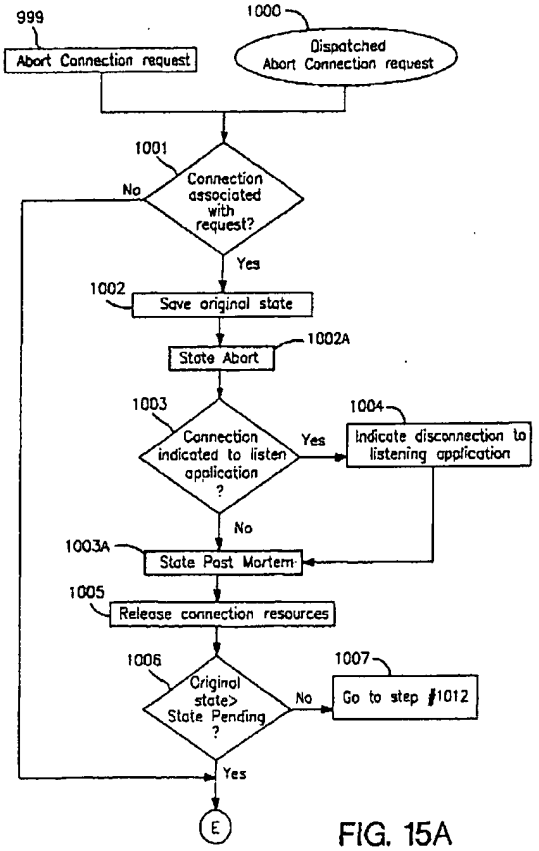


FIG. 15A

Abort Connection request logic

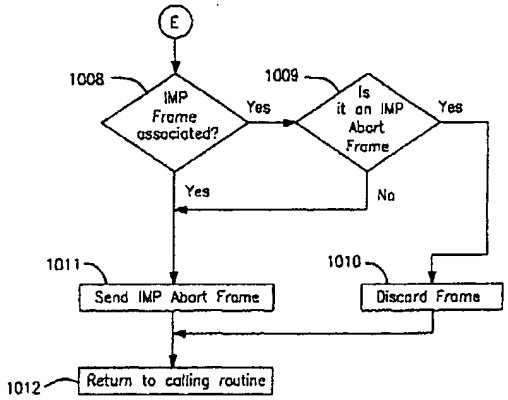

E
1008
IMP Frame associated?
Yes
1009
Is it an IMP Abort Frame
Yes
Yes
No
1011
Send IMP Abort Frame
1010
Discard Frame
1012
Return to calling routine

FIG. 15B

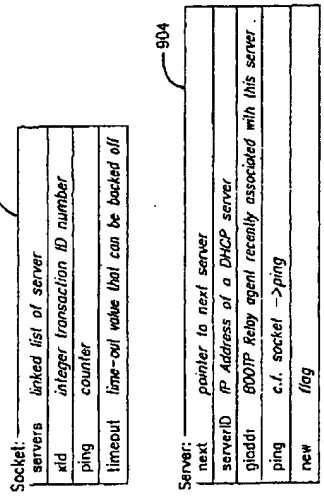


FIG. 16
DHCP Listener Data Structures

Fig. 17

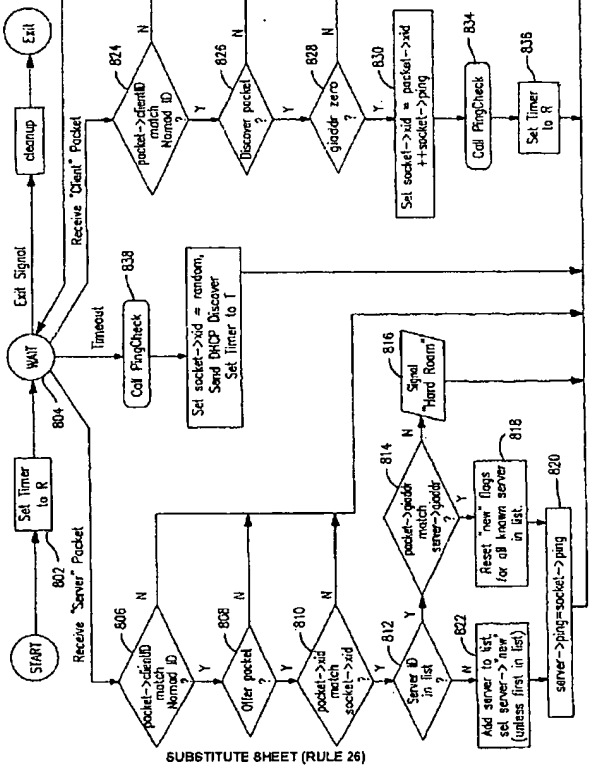
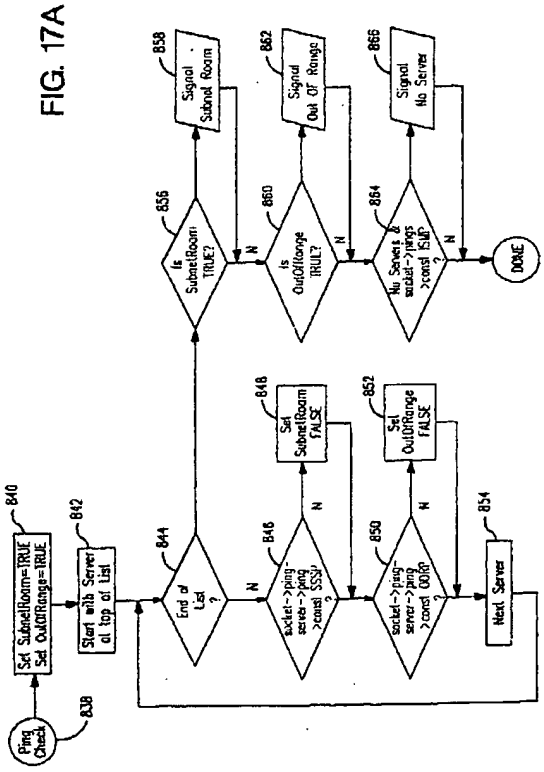

START
802
Set Timer to R
Receive "Server" Packet
804
WAIT
Exit Signal
Receive "Client" Packet
Timeout
cleanup
Exit
806
packet->clientID match Nomad ID ?
808
Offer packet ?
810
packet->xid match socket->xid ?
812
Server ID in list ?
814
packet->yiaddr match server->yiaddr ?
816
Signal "Hard Roam"
818
Reset "new" flags for all known servers in list.
820
server->ping=socket->ping
822
Add server to list, set server->"new" (unless first in list)
824
packet->clientID match Nomad ID ?
826
Discover packet ?
828
yiaddr zero ?
830
Set socket->xid = packet->xid ++socket->ping
834
Call PingCheck
836
Set Timer to R
838
Call PingCheck
Set socket->xid = random, Send DHCP Discover Set Timer to T

FIG. 17A


Ping Check
838
Set SubnetRoom=TRUE
Set OutOfRange=TRUE
840
Start with Server at top of List
842
End of List ?
844
Socket->ping-server->ping >const SSSU ?
846
Set SubnetRoom FALSE
848
Socket->ping-server->ping >const OORP ?
850
Set OutOfRange FALSE
852
Next Server
854
Is SubnetRoom= TRUE?
856
Signal Subnet Room
858
Is OutOfRange TRUE?
860
Signal Out Of Range
862
No Servers & socket->pings >const ICMP ?
864
Signal No Server
866
DONE
N

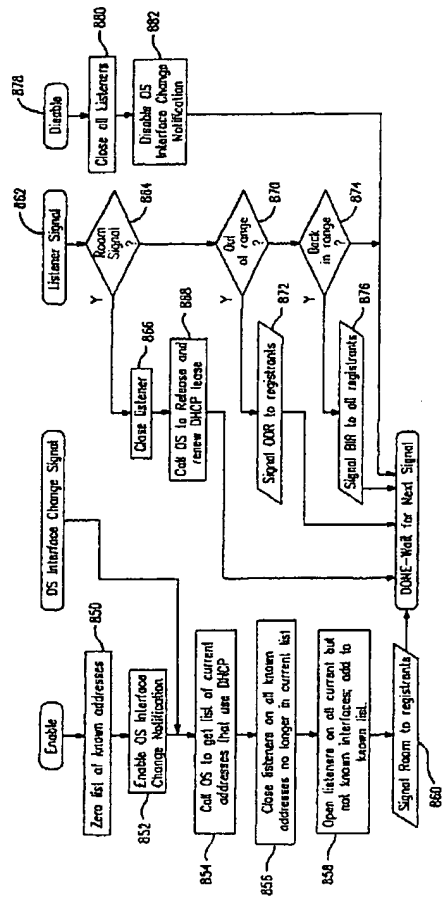


FIG. 18
ROAMING CONTROL CENTER-
Mobile End System

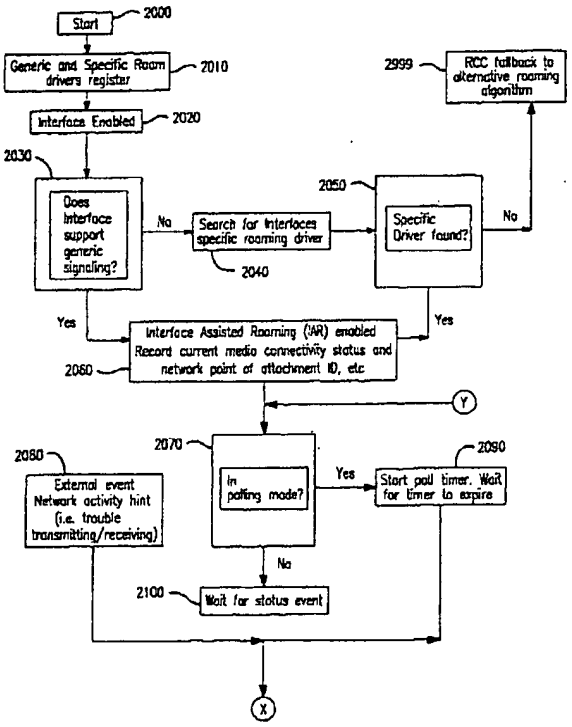
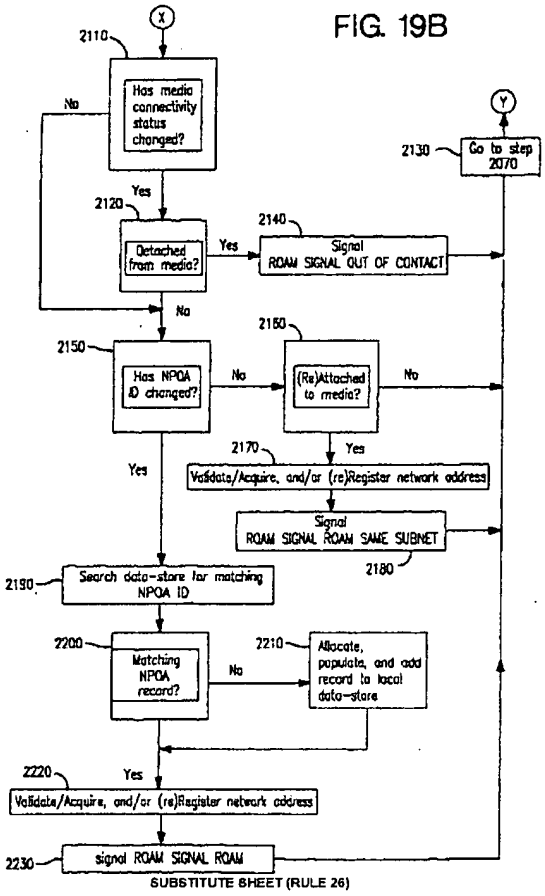


FIG. 19A

Interface Assisted Roaming (IAR) Decision Tree

FIG. 19B


X
2110
Has media connectivity status changed?
No
Yes
2120
Detached from media?
Yes
2140
Signal ROAM SIGNAL OUT OF CONTACT
No
2130
Go to step 2070
Y
2150
Has NPQA ID changed?
No
2160
(Re)Attached to media?
No
Yes
2170
Validate/Acquire, and/or (re)Register network address
Signal ROAM SIGNAL ROAM SAME SUBNET
2180
Yes
2190
Search data-store for matching NPQA ID
2200
Matching NPQA record?
No
2210
Allocate, populate, and add record to local data-store
2220
Yes
Validate/Acquire, and/or (re)Register network address
2230
signal ROAM SIGNAL ROAM

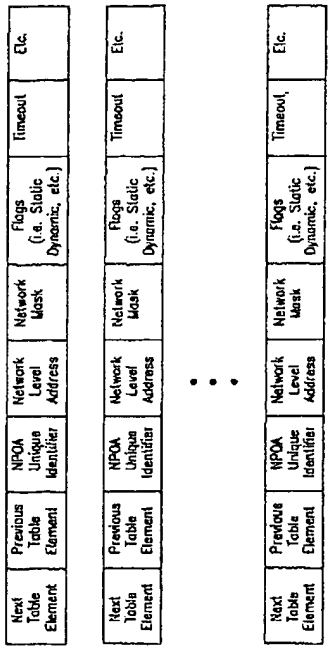


FIG. 20
Interface Assisted Roaming
Topology Node

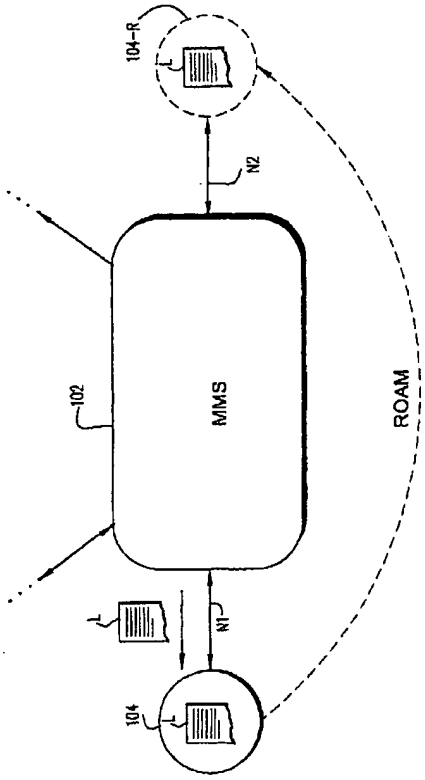


Fig.21
Disjoint network Roaming

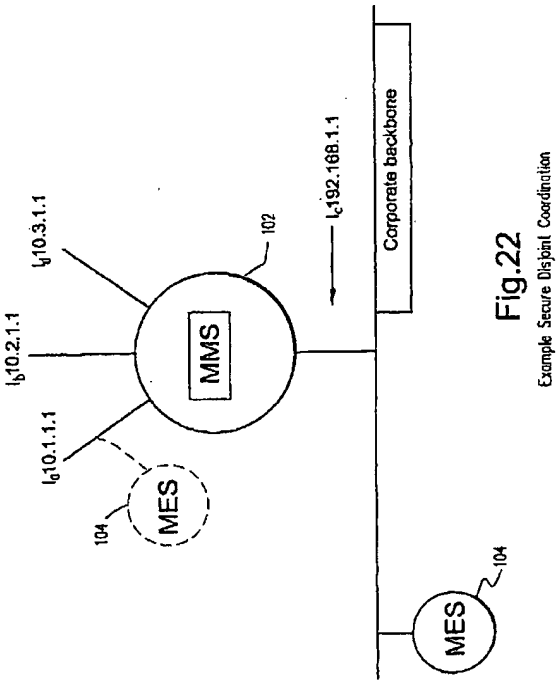


Fig.22
Example Secure Disjoint Coordination

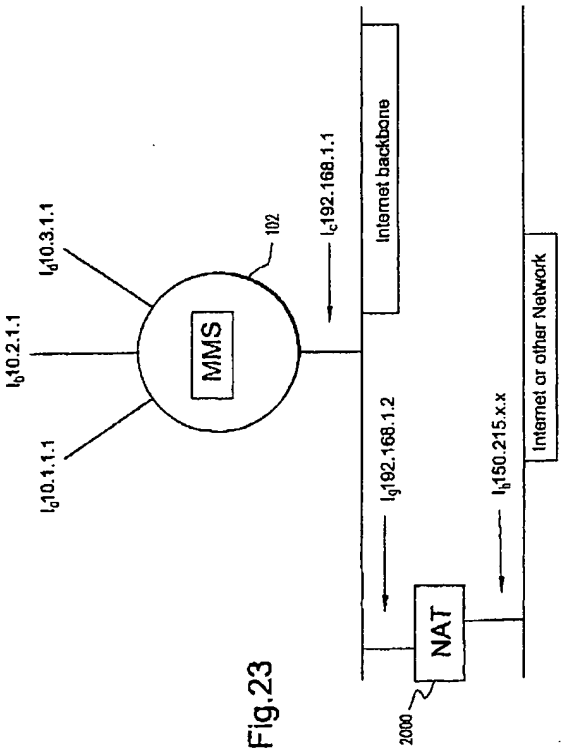

I_d10.3.1.1
I_b10.2.1.1
I_g10.1.1.1
MMS
102
I_c192.168.1.1
Internet backbone
I_g192.168.1.2
I_h150.215.x.x
Internet or other Network
NAT
2000

Fig.23

Example Policy Management Rules Table

TX/RX	Proxied	MES Source Port	MES Source Address	MES Dest Port	MES Dest Address	BPS (Available)	Process Name	Network	Location (GPS Coordinates)	Network Point of Attachment	User	Deny Request
T/R	Y	Any	Any	21	Any	< 100,000	Any	Any	Any	Any	US Patent Office	Y
T/R	Y	Any	Any	20	Any	< 100,000	Any	Any	Any	Any	US Patent Office	Y
T	N	5008	Any	5008	10.1.1.1		Any	Any	Any	Any	US Patent Office	N
R	N	5008	10.1.1.1	5008	Any		Any	Any	Any	Any	US Patent Office	N

Assumptions
1. Peer File Transfer Protocol control and data ports are 21 and 20
2. * indicates wildcard
3. MMS network address and port is 10.1.1.1: 5008
4. MES network port that frames from MMS is received on is 5008

In the example above all connections to destination ports 20 and 21 are denied or throttled if the available bandwidth is reduced to less then 100,000 bytes per second. In this example rules (rows) 3 and 4 only allow network traffic to flow to and from the MMS. All other network traffic that is not proxied is implicitly discarded. It should be appreciated that this table does not represent the full set of metrics that can be defined for policy management. Others variables such as monetary cost, location, network point of attachment, etc. can be added to the decisio tree. Furthermore, the rules engine interpreting theso entries can be distributed between the MES and MMS. As such either side or both may enforce the specified policy.

Fig.24

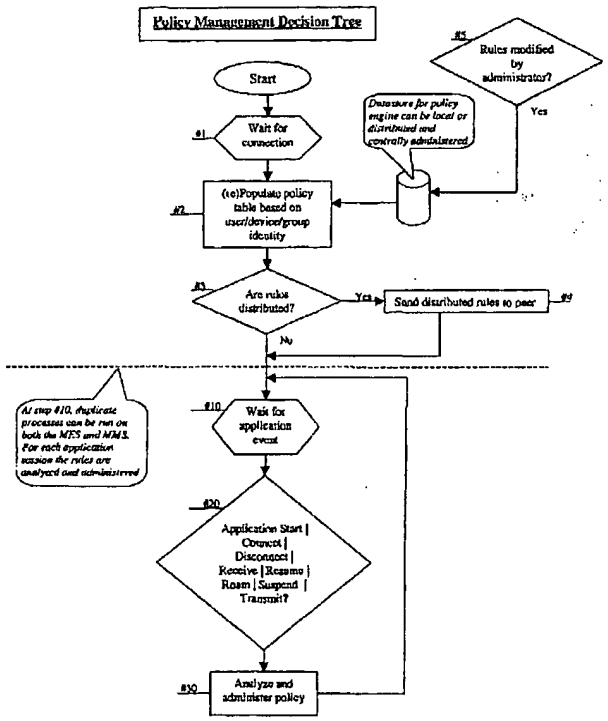

Policy Management Decision Tree
Start
#1
Wait for connection
#2
(re)Populate policy table based on user/device/group identity
#3
Are rules distributed?
Yes
Send distributed rules to peer
#4
No
#5
Rules modified by administrator?
Yes
Datastore for policy engine can be local or distributed and centrally administered
At step #10, duplicate processes can be run on both the MES and MM3. For each application session the rules are analyzed and administered
#10
Wait for application event
#20
Application Start | Connect | Disconnect | Receive | Resume | Roam | Suspend | Transmit?
#30
Analyze and administer policy

Fig.25

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/45581

A. CLASSIFICATION OF SUBJECT MATTER
(IPC(7)) : G06F 15/16
US CL : 709/227, 230, 229, 246, 310; 455/445, 414; 370/349, 715/500, 513
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
US : 709/227, 230, 229, 246, 310; 455/445, 414; 370/349; 715/500, 513

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EAST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y,P	US 6,167,513 A(INOUE et al) 26 December 2000, abstract, col. 4 lines 34-40, col. 14 lines 30-47, cols. 15-16 lines 20-35, cols. 17-18 lines 66-22	1-125
Y,P	US 6,170,057 B1 (INOUE et al) 02 January 2001, col. 7 lines 38-54, col. 9 lins 8-21	1-10
Y	US 6,006,090 A(COLEMAN et al) 21 December 1999, col. 1 lines 8-10, col. 2 lines 42-52 col. 3 lines 1-8, 14-20 and 58-65, col. 4 lines 3-20, col. 52 lines 1-5	11-30, 106-125
Y,P	US 6,147,986 A(ORSIC) 14 November 2000, abstract, col. 1 lines 43-59, col. 2 lines 13-67, col. 3 lines 1-57, cols. 5-8 lines 21-12	31-105

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 10 JANUARY 2002	Date of mailing of the international search report 28 JAN 2002
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer AYAZ SHEIKH Telephone No. (703)305-9600

Form PCT/ISA/210 (second sheet) (July 1998)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/22501

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y,P	US 6,230,004 B1 (HALL et al) 08 May 2001, col.1 lines 30-37 and 66-67, col. 2 lines 1-50)	98-105
Y,P	US 6,249,818 B1 (SHARMA) 19 June 2001, abstract, col. 1 lines 21-47, col. 2 lines40-48, col. 3 lines 33-57	98-105

Form PCT/ISA/210 (continuation of second sheet) (July 1998)*

フロントページの続き

(81)指定国 AP(GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW

(特許庁注：以下のものは登録商標)
Windows
ポケットベル

(72)発明者 ハンソン，アーロン，ディー.
アメリカ合衆国，９８１０７ ワシントン州，シアトル，ノースウェスト シックスティーサード ストリート ３００２
(72)発明者 スタニオロ，エミル，エー.
アメリカ合衆国，４４２５６ オハイオ州，メディーナ，アラメダ コート ４０５０
(72)発明者 メン，アナトリー
アメリカ合衆国，９８１３３ ワシントン州，シアトル，ナンバー４，ノース アハンドレッドセブンティーフィフス ８１６
(72)発明者 オルソン，エリック，ディー.
アメリカ合衆国，９８１１７ ワシントン州，シアトル，エヌダブリューエス エイティーセカンド ストリート ３０６
(72)発明者 サヴァレセ，ジョセフ，ティー.
アメリカ合衆国，９８０２０ ワシントン州，エドモンズ，ナインティーフィフス プレイス ウェスト ２２２０５

Fターム(参考) 5B085 BA06 BC01 BG02 BG07
5B089 GB01 KB04
5K033 CB06 CB08 CC01 DA17 DB16 DB18

【要約の続き】
ィ管理サーバは、非接続のネットワーク上で該サーバとコンタクトをとるためのリストを、モバイル端末システムに配布する。